THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*

Présentée et soutenue le *31/10/2015* **par :**

Sylvain MERCIER

Fast nonlinear solvers in solid mechanics.

JURY

Serge GRATTON	Professeur	INP Toulouse
Patrick HILD	Professeur	Université Paul Sabatier
Frédéric MAGOULÈS	Professeur	École Centrale Paris
Christian REY	Professeur	SAFRAN Tech
Bruno SCHEURER	Directeur de Recherche	CEA
Nicolas TARDIEU	Ingénieur de Recherche	EDF R&D
Xavier VASSEUR	Chercheur	CERFACS

École doctorale et spécialité :
MITT : Domaine Mathématiques : Mathématiques appliquées
Unité de Recherche :
CERFACS (CERFACS)
Directeur(s) de Thèse :
Serge GRATTON et *Xavier VASSEUR*
Rapporteurs :
Frédéric MAGOULÈS et *Christian REY*

CERFACS
42 Avenue Gaspard Coriolis,
31057 TOULOUSE Cedex 01,
France.

EDF R&D
1 avenue du général de Gaulle,
92141 CLAMART Cedex 05,
France.

Résumé

Résumé

La thèse a pour objectif le développement de méthodes performantes pour la résolution de problèmes non-linéaires en mécanique des solides. Il est coutume d'utiliser une méthode de type Newton qui conduit à la résolution d'une séquence de systèmes linéaires. De plus, la prise en compte des relations linéaires imposées à l'aide de multiplicateurs de Lagrange confère aux matrices une structure de point-selle.
Dans un cadre plus général, nous proposons, étudions et illustrons deux classes d'enrichissement de préconditioneurs (limited memory preconditioners) pour la résolution de séquences de systèmes linéaires par une méthode de Krylov. La première est une extension au cas symétrique indéfini d'une méthode existante, développée initialement dans le cadre symétrique défini positif. La seconde est plus générale dans le sens où elle s'applique aux systèmes non symétriques. Ces deux familles peuvent être interprétées comme des variantes par blocs de formules de mise à jour utilisées dans différentes méthodes d'optimisation. Ces techniques ont été développées dans le logiciel de mécanique des solides Code_Aster (dans un environnement parallèle distribué via la bibliothèque PETSc) et sont illustrées sur plusieurs études industrielles. Les gains obtenus en terme de coût de calcul sont significatifs (jusqu'à 50%), pour un surcoût mémoire négligeable.

Mots-clefs

Mécanique des solides, Itérations de Newton, Systèmes point selle, Préconditionneurs à mémoire limitée, Vecteurs de Ritz (harmonique)

Abstract

Abstract

The thesis aims at developing efficient numerical methods to solve nonlinear problems arising in solid mechanics. In this field, Newton methods are currently used, requiring the solution of a sequence of linear systems. Furthermore, the imposed linear relations are dualized with the Lagrange multiplier method, leading to matrices with a saddle point structure.
In a more general framework, we propose two classes of preconditioners (named limited memory preconditioners) to solve sequences of linear systems with a Krylov subspace method. The first class is based on an extension of a method initially developed for symmetric positive definite matrices to the symmetric indefinite case. The second class handles the more general case related to nonsymmetric matrices. Both families can be interpreted as block variants of updating formulas used in numerical optimization. They have been implemented into the Code_Aster solid mechanics software (in a parallel distributed environment using the PETSc library). These new preconditioning strategies are illustrated on several industrial applications. We obtain significant gains in computational cost (up to 50%) at a marginal overcost in memory.

Keywords

Solid mechanics, Newton iterations, Saddle point systems, Limited memory preconditioners, (Harmonic) Ritz vectors

Remerciements

Ce travail a été réalisé dans le cadre d'une convention CIFRE, entre la direction R&D d'EDF et le laboratoire du CERFACS. J'adresse ainsi mes premiers remerciements à l'ensemble des personnes ayant contribué à l'élaboration de ce projet de thèse ainsi qu'à l'Agence Nationale de la Recherche et de la Technologie (ANRT).

Je tiens à remercier sincèrement mes encadrants de thèse. D'une part, merci à mes directeurs Serge Gratton et Xavier Vasseur qui m'ont chaleureusement accueilli au sein de l'équipe Algo du CERFACS. Ils m'ont guidé et m'ont aidé de par leurs larges connaissances scientifiques et leur pédagogie tout au long de cette aventure. Merci également à mon encadrant industriel Nicolas Tardieu pour l'attention qu'il m'a témoignée durant ces trois années et pour la confiance qu'il m'a accordée en me proposant ce sujet de thèse. Sa connaissance de Code_Aster et sa vision de la recherche industrielle en général m'ont été d'une grande aide.

Au delà de mes encadrants, j'ai pu côtoyer et échanger avec de nombreuses personnes, que ce soit au sein de l'équipe Algo du CERFACS ou du département AMA d'EDF R&D. Leurs conseils et leur soutien m'ont aussi permis de mener à bien ces travaux de thèse. Je remercie en particulier Thomas de Soza que j'ai sollicité à de nombreuses reprises et qui m'a toujours aidé avec enthousiasme et efficacité, de même que mon collègue de bureau Guillaume Drouet avec qui j'ai pu partager les joies et les galères d'une telle expérience.

Je remercie Frédéric Magoulès et Christian Rey pour le temps consacré à lire puis rapporter ce manuscrit, ainsi que pour leur participation à mon jury de thèse. Je suis de même très reconnaissant envers Patrick Hild et Bruno Scheurer qui ont accepté d'assister à ma soutenance et de faire partie de mon jury de thèse.

Merci à ma famille qui m'a toujours encouragé et soutenu dans mes choix. Merci à mes amis (de Paris, Bretagne, Vendée et ailleurs) qui m'accompagnent depuis longtemps. Pour finir, j'ai une pensée toute particulière pour Quentin avec qui j'aurais aimé partager cette réussite.

Contents

List of Figures **13**

List of Tables **15**

Introduction **21**

1 Framework and mathematical background **25**
1.1 Solid mechanics framework 25
1.1.1 Linear model problem and boundary conditions 25
1.1.2 Nonlinearities in solid mechanics 29
1.1.3 Integration within the Newton's method 32
1.2 Krylov subspace methods 34
1.2.1 Presentation 34
1.2.2 The GMRES method 37
1.2.3 Ritz and harmonic Ritz pairs 42
1.2.4 Preconditioning 45
1.3 Linear systems of saddle point structure 48
1.3.1 Segregated solvers 49
1.3.2 Coupled solvers 51
1.4 Conclusions 56

2 Limited memory preconditioners for symmetric indefinite systems **57**
2.1 Limited memory preconditioners for symmetric definite matrices 58
2.1.1 Limited memory preconditioners for symmetric positive definite matrices 58
2.1.2 Limited memory preconditioners for symmetric negative definite matrices 61
2.2 Limited memory preconditioners for symmetric indefinite matrices 62

2.2.1 Definition . . . 62
2.2.2 Spectrum of AH . . . 63
2.2.3 Sign of the eigenvalues of AH and inertia of H . . . 64
2.2.4 Nonexpansion of the spectrum of AH . . . 66
2.2.5 Ritz limited memory preconditioner . . . 71
2.3 Implementation considerations . . . 76
2.3.1 Computational cost and memory requirements for a general matrix S 76
2.3.2 Computational cost and memory requirements of the Ritz-LMP$_{\pm}$. 77
2.3.3 Case of harmonic Ritz vectors as columns of S . . . 78
2.4 Applications to solid mechanics . . . 79
2.4.1 Sequence of preconditioned saddle point systems . . . 79
2.4.2 Mechanical bearing (linear case) . . . 81
2.4.3 Containment building of a nuclear reactor . . . 84
2.4.4 Mechanical bearing (nonlinear case) . . . 87
2.4.5 Polycrystalline aggregate . . . 89
2.5 Conclusions . . . 92

3 Limited memory preconditioners for nonsymmetric systems 93
3.1 Motivations and definition . . . 94
3.2 Spectral characterization of the preconditioned matrix . . . 98
3.3 Theoretical comparison with existing methods . . . 100
3.3.1 Deflation . . . 101
3.3.2 Abstract balancing preconditioner . . . 102
3.3.3 Spectral comparison . . . 103
3.3.4 Comparison of GMRES iterates . . . 104
3.4 Implementation considerations . . . 110
3.4.1 Choice of the matrix S . . . 110
3.4.2 Computational cost and memory requirements of the LMP$_{ns}$ . . . 110
3.4.3 Computational cost and memory requirements of the deflation . . 111
3.5 Applications to solid mechanics . . . 112
3.5.1 Sequence of preconditioned saddle point systems . . . 112
3.5.2 Mechanical bearing (linear case) . . . 115
3.5.3 Containment building of a nuclear reactor . . . 118
3.5.4 Shut down nuclear reactor cooling loop . . . 120
3.5.5 Snap hook . . . 123
3.6 Conclusions . . . 125

Conclusions & Perspectives **127**

A Description of the code **133**

A.1 Block preconditioners for saddle point systems 133

A.2 Limited memory preconditioners 135

A.2.1 Development in PETSc 135

A.2.2 Development in Code_Aster 138

Bibliography **141**

List of Figures

1.1 Steam generator working 30
1.2 Picture of a plug and a cut of a tube 30
1.3 Mesh of the bottom of a steam generator. 31
1.4 Model of a plug insertion in a tube. 31
1.5 Constraint field after the plug insertion. 31

2.1 Eigendistribution of A and AH: case of $1 \in [\sigma_1, \sigma_N]$. 60
2.2 Eigendistribution of A and AH: case of eigenvalues of A larger than 1. . . 61
2.3 Eigendistribution of A and AH: case of $1 \in [\sigma_1^+, \sigma_{N_+}^+]$. 70
2.4 Eigendistribution of A and AH: case of positive eigenvalues of A larger than 1. 71
2.5 Mesh of the mechanical bearing. 81
2.6 Mechanical bearing: spectrum of $\mathcal{A}$. 82
2.7 Mechanical bearing (linear case): convergence history of GMRES(30). Three preconditioning methods are compared: no second level preconditioning, limited memory preconditioners based on $k = 5$, 20 or 30 Ritz vectors and harmonic Ritz vectors, respectively. 83
2.8 Mechanical bearing (linear case): convergence history of GMRES(30). Four preconditioning methods are compared: no second level preconditioning, limited memory preconditioners based on $k = 5$, 20 or 30 Ritz vectors, projected Ritz vectors and exact eigenvectors, respectively. . . . 84
2.9 Containment building: three-dimensional mesh (left part) and location of the prestressing tendons on the surface (right part). 85
2.10 Containment building: convergence history of preconditioned GMRES(30) for the last three linear systems in the sequence. Case of limited memory preconditioners with $k = 5$, 20 or 30 Ritz vectors. 86

2.11 Mechanical bearing (nonlinear case): convergence history of preconditioned GMRES(30) for the last three linear systems in the sequence. Case of limited memory preconditioners with $k = 5$, 20 or 30 Ritz vectors. . . . 88
2.12 Polycrystalline aggregate: unstructured mesh of the representative elementary volume (left part) and detailed view of some grains of the polycrystalline aggregate (right part). 89
2.13 Polycrystalline aggregate: sketch of the boundary conditions. 90
2.14 Polycrystalline aggregate (fine mesh calculation): GMRES(30) convergence history for different preconditioners at 2 different iterations of the Newton's method (2nd and 14th iterations). 91

3.1 Mechanical bearing: spectrum of $\mathcal{A}$. 115
3.2 Mechanical bearing: convergence history of GMRES(30). Four preconditioning methods are compared: no second level preconditioner, limited memory preconditioners based on $k = 5$, 20 or 30 Ritz vectors, harmonic Ritz vectors and exact eigenvectors. 117
3.3 Mechanical bearing: convergence history of GMRES(30). Three methods are compared, all based on $k = 5$, 20 or 30 Ritz vectors: limited memory preconditioner, balancing preconditioner and deflation. 118
3.4 Containment building: convergence history of preconditioned GMRES(30) for the last three linear systems in the sequence. Case of limited memory preconditioners with $k = 5$, 20 or 30 Ritz vectors. 120
3.5 Mesh of the shut down nuclear reactor cooling loop. 121
3.6 Shut down nuclear reactor cooling loop: speedup curve 123
3.7 Mesh of the snap hook. 124

List of Tables

2.1 Containment building: cumulative iteration count for the last three linear systems in the sequence, CPU time and memory requirements for different limited memory preconditioners. Case of $k = 5$, 20 or 30 Ritz vectors. . . 87

2.2 Mechanical bearing (nonlinear case): cumulative iteration count for the last three linear systems in the sequence, CPU time and memory requirements for different limited memory preconditioners. Case of $k = 5$, 20 or 30 Ritz vectors. 89

2.3 Polycrystalline aggregate: cumulative iteration count over the complete Newton's sequence, CPU time and memory requirements for different preconditioners. Results are given for three different levels of mesh refinement (coarse, intermediate and fine, respectively). 91

3.1 Containment building: cumulative iteration count for the last three linear systems in the sequence, CPU time and memory requirements for different limited memory preconditioners. Case of $k = 5$, 20 or 30 Ritz vectors. . . 119

3.2 Containment building: cumulative iteration count for the last three linear systems in the sequence, CPU time and memory requirements for different deflation methods. Case of $k = 5$, 20 or 30 Ritz vectors. 119

3.3 Shut down nuclear reactor cooling loop: cumulative iteration count over the complete Newton's sequence, CPU time and memory requirements for limited memory preconditioner and deflation with $k = 5$ Ritz vectors. . . 122

3.4 Snap hook: cumulative iteration count over the complete Newton's sequence, number of required $LDLT\_SP$ factorizations, CPU time and memory requirements for limited memory preconditioner and deflation with $k = 5$ Ritz vectors. 125

List of Algorithms

1 Arnoldi method to compute an orthonormal basis of $\mathcal{K}_l(A, r_0)$ 36
2 Main steps of GMRES to solve $Ax = b$ 38
3 GMRES algorithm to solve $Ax = b$ 40
4 GMRES(m) algorithm with a split preconditioner 47
5 Compute Z, Y and Σ such that $H = (I_N - ZY^T)(I_N - YZ^T) + Z\Sigma Z^T$ . . 77
6 Application of the limited memory preconditioner: $r = Hx$ 77
7 Application of the Ritz limited memory preconditioner: $r = Hx$ 78
8 Class of Broyden methods to solve $Ax = b$, using an approximation of A^{-1} 94
9 EN-like methods to solve $Ax = b$, using an approximation of A^{-1} 95
10 Deflated GMRES(m) to solve $Ax_D = b$ 108
11 GMRES(m) with a right balancing preconditioner to solve $Ax_B = b$ . . . 109
12 Compute Y, $X \in \mathbb{R}^{N\times k}$ such that $H_{ns} = I_N + YX^T$ 111

Notations

$\|.\|_2$	Euclidean norm
$\|.\|_F$	Frobenius norm
I_k	Identity matrix of dimension k
$0_{i,j}$	Zero rectangular matrix with i rows and j columns. The notation 0 is used if there is no ambiguity
$\Lambda(A)$	Spectrum of A
$\mathcal{R}(A)$	Range of a given matrix A
$\mathcal{N}(A)$	Null space of a given matrix A
$P_{\mathcal{V},\mathcal{W}}$	Projection operator onto $\mathcal{V}$ along $\mathcal{W}$, where $\mathcal{V}$ and $\mathcal{W}$ denote complementary subspaces of a vector space. $P_{\mathcal{V},\mathcal{W}}$ is the unique projection operator with range $\mathcal{R}(P_{\mathcal{V},\mathcal{W}}) = \mathcal{V}$ and null space $\mathcal{N}(P_{\mathcal{V},\mathcal{W}}) = \mathcal{W}$
$P_{\mathcal{V}}$	Orthogonal projection operator onto $\mathcal{V}$
$A\|_{\mathcal{S}}$	Restriction of the matrix A to the subspace $\mathcal{S}$

Introduction

The numerical solution of large-scale industrial scientific problems generally requires large computational times and large memory needs. For many years, many efforts have concerned both computer architecture aspects and the design of efficient algorithms, in order to provide effective simulation tools. In this thesis, we are rather interested in the second aspect, whose choice can be justified by the following quote from Philippe Toint: "I would rather have today's algorithms on yesterday's computers than vice versa". Even so, we want to propose new techniques well adapted to current computer science tools, such as multiprocessing and parallel computing.

Among several domains of application, the PhD thesis focuses on the solution of problems issued from solid mechanics. More specifically, the main goal is to provide efficient methods to speedup the solution of nonlinear problems discretized within the framework of the open-source software Code_Aster [1], which is a general purpose finite element code developed at EDF (Electricité de France). For more than 20 years, Code_Aster serves as the simulation tool used by the engineering departments of EDF to analyze the various components of the electricity generation facilities and to produce safety analysis. As we will see in Chapter 1, this industrial software mostly relies on Newton's method to solve nonlinear problems, leading to a sequence of linear systems of the form

$$A_i x_i = b_i, \quad \text{for } i = 1, \cdots, I, \tag{1}$$

where x_i, $b_i \in \mathbb{R}^N$ and $A_i \in \mathbb{R}^{N \times N}$ are sparse and are assumed to slowly vary all along the sequence. Furthermore, these matrices have a saddle point structure (due to the dualization of different types of conditions) and are symmetric indefinite.

Two main families can be used to solve such linear systems: sparse direct factorization [27] and iterative methods [90]. The first approach consists in factorizing the given matrix as a product of triangular matrices (e.g. with a sparse LU factorization), then in

computing a solution thanks to successive forward and backward substitutions. In solid mechanics, as in other applications, these methods have proved to be efficient in the two-dimensional case, but a significant fill-in phenomenon usually occurs when factorizing large-scale three-dimensional problems [11, 99]. Hence, the memory requirement generally compromises their use, even more when each left-hand side in (1) needs to be factorized (i.e. when the matrices change all along the sequence). On the other hand, iterative methods provide a sequence of (improving) approximations of the solution of the linear system. Contrary to the direct solvers, these methods only require the knowledge of the action of the matrix on a vector. Furthermore, they are particularly well adapted to parallel computing (see, e.g., Chapter 11 in [90]). Among all these methods available in the literature, it is known that Krylov subspace methods are the method of choice for large-scale problems, especially when solving sequences as (1) in the case where left-hand sides are changing [90]. However, Krylov subspace methods are generally efficient when they are are combined with preconditioning [11]. This concept, whose design remains an active domain of research, consists conceptually in multiplying the matrix of the original system by another matrix called preconditioner, while maintaining the same solution. The aim is to obtain a new operator with better properties (detailed later); the preconditioner generally corresponds either to an approximation of the inverse of the original matrix or to the inverse of an approximation of the original matrix. It is worth mentioning that the computation of a preconditioner can be rather expensive, and in the context of the solution of (1) with slowly varying matrices, a preconditioner is often reused for several successive linear systems.

Beyond solid mechanics, the numerical solution of sequences of linear systems is frequently required in many applications in computational science and engineering. Using the fact that the operators in subsequent linear systems have most often similar spectral properties, a first possible approach to design efficient numerical methods is to extract information generated during the solution of a given linear system to improve the convergence rate of the Krylov subspace method during the subsequent solutions. Deflated and augmented Krylov subspaces [25, 29, 77, 92] or Krylov subspace methods with recycling [60, 83, 101, 112] have been proposed in this setting. We refer the reader to [38, 39, 51, 99] for a comprehensive theoretical overview on these methods and to references therein for a summary of applications, where the relevance of these methods has been shown. An alternative consists in exploiting information generated during the solution of a given linear system to improve a preconditioner when solving the next linear system in the sequence. This is the main subject that we want to address in this

thesis. Several contributions already exist in the literature in different domains of application, such as [12, 46, 34] for symmetric positive definite systems, or [10, 106] when the successive left-hand sides are nonsymmetric. In this dissertation, we propose two preconditioning improvement methods issued from the numerical optimization literature, which allow us to improve the effect of an existing first-level preconditioner. The definition of these new preconditioners involves only a small number k of vectors (that may satisfy some assumption), and usually requires the product of a matrix A_i with these vectors. In particular, we study preconditioners based on the selection of approximate eigenvectors obtained during the solution of a given linear system, with application on real-life numerical experiments.

This thesis is organized as follows: Chapter 1 introduces fundamental information that will be used later in the other chapters. We start by explaining how a nonlinear solid mechanics problem, handled within Code_Aster, leads to the solution of a sequence of symmetric linear systems with saddle point structure. Then, we give relevant information related to Krylov subspace methods (and particularly preconditioned GMRES). We explain the notion of (harmonic) Ritz pairs corresponding to approximations of eigenvectors as well. Finally, we give a brief overview of existing methods for the solution of linear systems with saddle point structure, where we notably introduce two block preconditioners which are used as first-level preconditioners later. We intend to make use of this opportunity to present the methods that are currently available in Code_Aster to solve nonlinear problems. This chapter does not provide any new results but simply aims at explaining the purpose of this thesis.

Chapter 2 details the first class of new preconditioners introduced in this thesis, denoted $\text{LMP}_{\pm}$, well adapted to the solution of sequences of symmetric indefinite linear systems as (1). This technique can be seen as an extension of limited memory preconditioners initially proposed in the symmetric positive definite case in [46]. This chapter is composed of two main parts. First, we analyse theoretically and characterize the spectral effect of the preconditioners on symmetric indefinite systems. More precisely, several results are given, related to different choices of the k vectors defining the $\text{LMP}_{\pm}$. The second part describes numerical experiments obtained within Code_Aster on large-scale applications in solid mechanics. In particular, we use the $\text{LMP}_{\pm}$ in addition to a symmetric positive definite first-level block diagonal preconditioner, taking into account the saddle point structure of the systems in (1). We will notice that the $\text{LMP}_{\pm}$ requires a symmetric definite positiveness property for the first-level preconditioner, a condition

that can be quite restrictive.

Hence, as a cure, we develop another class of limited memory preconditioners called LMP_{ns}, well suited to the solution of nonsymmetric linear systems. Although the saddle point matrices in (1) are assumed to be symmetric indefinite in this manuscript, this new class does not require any assumption on the first-level preconditioner. Chapter 3 first explores some properties of the LMP_{ns}. We characterize the spectrum of the preconditioned matrix, and we further carry out a comparison of the LMP_{ns} with both deflation [39] and the abstract balancing preconditioner [33]. More precisely, we compare the respective spectra and GMRES iterates (including also the $\text{LMP}_{\pm}$ in a more general framework). Then, we illustrate the numerical efficiency of the LMP_{ns} on possibly large-scale applications in solid mechanics, using as a first-level preconditioner either a block upper triangular preconditioner or the default preconditioning method available in Code_Aster.

Then, we draw both final remarks and future research plans in Section "Conclusions & Perspectives".

In Appendix, we finally provide brief information about the implementation of the proposed preconditioning methods. In fact, the nonlinear process for the solution of various solid mechanics problems is handled by Code_Aster, but we rely on the PETSc library [3] to solve the successive linear systems in (1). Thus, part of the code has been developed in PETSc and an interface with this library has been implemented in the solid mechanics Code_Aster software.

Chapter 1

Framework and mathematical background

This chapter gathers useful information of interest in the next chapters. Section 1.1 aims at introducing the solid mechanics problems that we want to solve in this thesis. First, we present a linear elastic problem and we show how to obtain the associated discretized linear problem from the finite element method, when some imposed conditions are dualized. Then, after discussing the different types of nonlinearities arising in solid mechanics, we present how the related mechanical problems can lead to a sequence of linear systems of saddle point structure, due to Newton's method.

In the case where both the left-hand sides and right-hand sides of the linear systems given in sequence are changing, it is known that preconditioned Krylov subspace methods are the method of choice, especially for large-scale problems. The presentation of these iterative techniques is the main purpose of Section 1.2, where the GMRES method is more specifically detailed. This latter method will be used in the following chapters and some fundamental information is given in relation with the future preconditioning approaches proposed in this manuscript.

The chapter ends with a brief survey of solution methods for saddle point linear systems. In particular, we present different block preconditioners to be used later.

1.1 Solid mechanics framework

1.1.1 Linear model problem and boundary conditions

As a first step, we focus on the equilibrium of an elastic body in solid mechanics, under the small displacement hypothesis. The following presentation is notably based on [21].

Let $\Omega \subset \mathbb{R}^d$, $d = 2, 3$, represent the reference configuration of an elastic body and $\partial\Omega = \Gamma_D \cup \Gamma_N$ be the boundary of this domain, where $\Gamma_D \cap \Gamma_N = \emptyset$. If the body is subject to volume loads $r \in (L^2(\Omega))^d$ in Ω, surface loads $h \in (L^2(\Gamma_N))^d$ on Γ_N, and a Dirichlet condition on Γ_D, the problem consists in finding the displacement field $u : \overline{\Omega} \to \mathbb{R}^3$ satisfying the governing equations:

$$-div(\sigma(u)) = r \text{ in } \Omega, \tag{1.1a}$$

$$\sigma(u)n = h \text{ on } \Gamma_N, \tag{1.1b}$$

$$u = u_d \text{ on } \Gamma_D, \tag{1.1c}$$

$$\sigma(u) = \mathcal{H}\epsilon(u), \tag{1.1d}$$

$$\epsilon(u) = (\nabla u + \nabla^T u)/2. \tag{1.1e}$$

The equalities (1.1a), (1.1b) and (1.1c) correspond to the equilibrium equation, the natural and the essential boundary conditions respectively, where σ is the stress tensor field and n is the outward unit normal to Ω on $\partial\Omega$. Furthermore, the constitutive law (1.1d) connects σ to the strain tensor field ϵ. In the elastic case, this latter relation is linear, and $\mathcal{H}$ is the fourth elastic coefficient tensor which satisfies both symmetry and ellipticity conditions and whose components are in $L^\infty(\Omega)$ (the Hooke's tensor). Finally, (1.1e) is known as the compatibility equation, linearized under the small displacement hypothesis.

1.1.1.1 The weak formulation with eliminated boundary conditions

A common weak formulation of this elastic problem is given by finding $u \in \mathcal{C}(u_d)$ such that

$$\int_\Omega \mathcal{H}\epsilon(u) : \epsilon(v)\ d\Omega = \int_\Omega rv\ d\Omega + \int_{\Gamma_N} hv\ d\Gamma \quad \forall v \in \mathcal{C}(0), \tag{1.2}$$

where

$$\mathcal{C}(u_d) = \{v \mid v \in (H^1(\Omega))^d \text{ and } v = u_d \text{ on } \Gamma_D\}$$

and

$$\mathcal{C}(0) = \{v \mid v \in (H^1(\Omega))^d \text{ and } v = 0 \text{ on } \Gamma_D\}.$$

$\mathcal{C}(u_d)$ and $\mathcal{C}(0)$ are the sets of admissible displacements with imposed and zero displacement, respectively. We refer the reader to [21] for details about how to obtain equality (1.2). Mechanically, it represents the principle of virtual work, illustrating the equality between the virtual work of the elastic forces (the left-hand side) and the external loads

(the right-hand side). A finite element method is commonly used to approximate the solution of the problem (1.2), but the description of the related process is out of the scope of this presentation (see e.g. [21, 35]). The important result comes from the associated linear system to solve, of the form

$$\tilde{G}\tilde{u} = \tilde{f}. \tag{1.3}$$

The matrix $\tilde{G} \in \mathbb{R}^{\tilde{n}\times\tilde{n}}$ is the stiffness matrix, where $\tilde{n}$ is the total number of unknowns (i.e. the dimension of the discretized subspace of $\mathcal{C}(u_d)$). $\tilde{u}$ and $\tilde{f} \in \mathbb{R}^{\tilde{n}}$ consist of discretized fields of displacement and nodal forces, respectively. By construction, the matrix $\tilde{G}$ is sparse and symmetric positive definite.

Remark 1. The equation (1.2) can also be obtained with a variational formulation, minimizing the potential energy [21]:

$$u = \underset{w\in\mathcal{C}(u_d)}{\operatorname{argmin}} \left(\frac{1}{2}\int_\Omega \mathcal{H}\epsilon(w) : \epsilon(w)\ d\Omega - \int_\Omega rw\ d\Omega - \int_{\Gamma_N} hw\ d\Gamma \right).$$

1.1.1.2 The weak formulation with dualized boundary conditions

However, in several industrial software like Code_Aster, the preferred formulation is written in a slightly different way. According to [21], we can use a different weak formulation, where the boundary conditions are dualized. This requires introducing the subsets

$$\mathcal{C} = \{v \mid v \in (H^1(\Omega))^d\}$$

and

$$\mathcal{C}'(\Gamma_D) = \{\mu \mid \int_{\Gamma_D} v\mu\ d\Gamma \text{ is defined } \forall v \in \mathcal{C}\}.$$

The associated weak formulation is then given by finding $(u, \lambda) \in \mathcal{C} \times \mathcal{C}'(\Gamma_D)$ such that

$$\begin{aligned} &\int_\Omega \mathcal{H}\epsilon(u) : \epsilon(v)\ d\Omega + \int_{\Gamma_D} \lambda v\ d\Gamma = \int_\Omega rv\ d\Omega + \int_{\Gamma_N} hv\ d\Gamma \quad \forall v \in \mathcal{C} \\ &\int_{\Gamma_D} u\mu\ d\Gamma = \int_{\Gamma_D} u_d\mu\ d\Gamma \quad \forall \mu \in \mathcal{C}'(\Gamma_D). \end{aligned} \tag{1.4}$$

A finite element discretization finally leads to solve a linear system of the form

$$\mathcal{K}x = b \Longleftrightarrow \begin{pmatrix} G & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix}, \tag{1.5}$$

where u and λ correspond now to both vectors of size n and m, respectively. n and m are the dimensions of the discretized subspaces for the displacement and the con-

straints, respectively. In this case, the matrix $G \in \mathbb{R}^{n\times n}$ is sparse and symmetric positive semidefinite. Indeed, the essential boundary conditions are not taken into account and the dimension of the kernel $\mathcal{N}(G)$ corresponds to the number of rigid body motions of the studied body [76]. For $d = 3$, this number is equal to 6, allowing 3 translations and 3 rotations. Here, the constraint matrix $B \in \mathbb{R}^{m\times n}$ has only one nonzero coefficient per row, since the essential boundary conditions are of the form $u_j = g_j$ where u_j is one component of u. They are also known as single-freedom constraints (SFCs).

Remark 2. In this case, the equations (1.4) can be obtained with the variational formulation:

$$(u, \lambda) = \underset{w\in\mathcal{C}}{\operatorname{argmin}} \underset{\mu\in\mathcal{C}'(\Gamma_D)}{\operatorname{argmax}} \left(\frac{1}{2} \int_\Omega \mathcal{H}\epsilon(w) : \epsilon(w)\ d\Omega - \int_\Omega rw\ d\Omega - \int_{\Gamma_N} hw\ d\Gamma + \int_{\Gamma_D} \mu(w - u_d)\ d\Gamma \right).$$

1.1.1.3 Other type of imposed conditions

The choice of the weak formulation with dualized boundary conditions in software like Code_Aster is justified by the need to impose other relations between degrees of freedom for some mechanical problems. As usually done in these industrial solvers, we introduce these relations directly into the discretized problem. They can model for instance a nondeformable part of the structure [84] or connecting conditions between modelings [70]. An example of this latter case will be introduced in Sections 2.4.3 and 3.5.3. These relations, known as multifreedom constraints (MFCs), generally connect two or more displacement components, written as

$$\sum_{i\in I} \beta_i u_i = \alpha, \quad \text{where } I \subset \{1, ..., n\}.$$

They can not be easily "eliminated" in the matrix G, contrary to the essential boundary conditions with the weak formulation (1.2), and are also dualized to obtain a linear system of saddle point structure like (1.5). Now, the matrix $B \in \mathbb{R}^{m\times n}$ represents m linear relations coming from both single and multifreedom constraints. B is supposed to be of full rank m, which states that the constraint conditions are non-redundant. Moreover, the matrix G is supposed to be positive definite on $\mathcal{N}(B)$, which ensures that the constraints forbid the rigid body motions of the structure.

Remark 3. The nonsingularity of the linear system (1.5) will be proved in Section 1.3.

Remark 4. In Code_Aster, the SFCs can be either directly "eliminated" or dualized.

Remark 5. Solving the saddle point system provides u and λ, and $-B^T\lambda$ can be interpreted as constraint forces arising from the essential boundary conditions and MFCs.

Remark 6. We emphasize the fact that the MFCs are added to the discretized problem in Code_Aster, as in many industrial software. This approach is known not to be optimal, contrary to the mortar approach whose use has also been extended to the contact problems [9]. Nevertheless, the imposition of MFCs directly on the discretized unknowns is much easier to implement and much more versatile (it easily mixes displacement and rotation) so that it is real common and powerfull modelization tool for the engineers.

1.1.2 Nonlinearities in solid mechanics

The linear case described above was particularly instructive to show how the different constraint conditions are treated. However, linear problems constitute just a part of the solid mechanics problems. Indeed, there are many different sources of nonlinearities in the response of structures under loading [20, 21, 5]. They can be grouped into three classes: material, geometric and contact nonlinearities.

As an illustration, we present an industrial study performed within Code_Aster, which is a good example of a nuclear safety analysis carried out at EDF. It deals with the insertion of a plug in a steam generator tube. In fact, in a Pressurized Water Reactor (PWR) nuclear power plant, a steam generator is a heat exchanger using the energy from the water of the primary circuit (heated by the nuclear reactor core) to transform the water of the secondary circuit into steam, in order to drive the turbine. Figure 1.1 presents a simplified drawing of a steam generator working. This component contains several thousand tubes, where the hot water of the primary circuit flows. These tubes are generally subject to variations of temperature and significant pressure, which can lead to the formation of cracks. This kind of defect is very critical, since the water of the secondary circuit could be contaminated by the fluid of the primary one. To remedy this problem, a plug can be inserted into the tube to take it out of service and insure the sealing (see Figure 1.2). Moreover, Figures 1.3 and 1.4 correspond to the meshes of the bottom of a steam generator and of a plug in a tube. More precisely, in Figure 1.4, the green part pictures the plug and the yellow one pictures the nut, used to fix the plug and removed at the end of the operation. Finally, Figure 1.5 presents the constraint field after the plug insertion, obtained within Code_Aster.

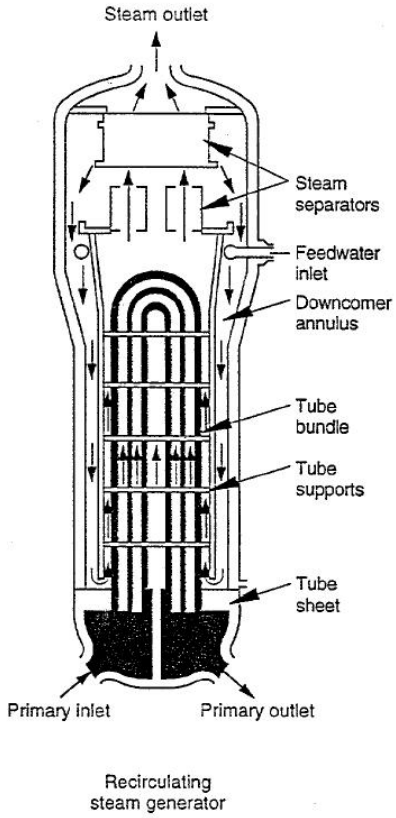

Figure 1.1: Steam generator working - Source: `http://www.allthingsnuclear.org`

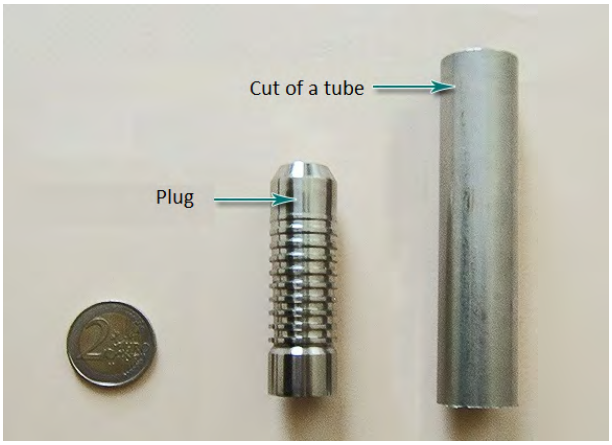

Figure 1.2: Picture of a plug and a cut of a tube - Source: `http://www.asn.fr`

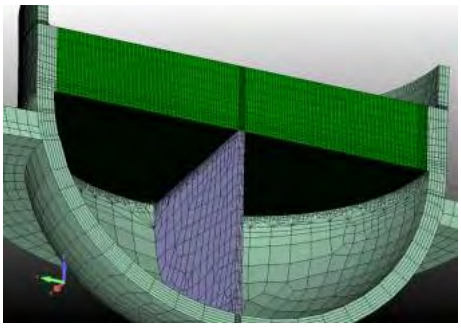

Figure 1.3: Mesh of the bottom of a steam generator.

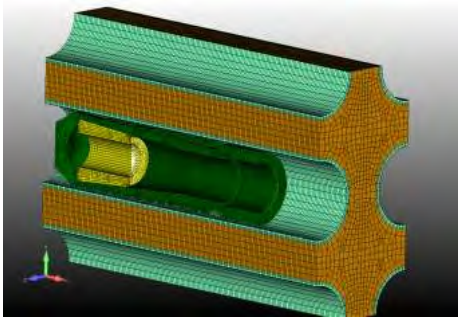

Figure 1.4: Model of a plug insertion in a tube.

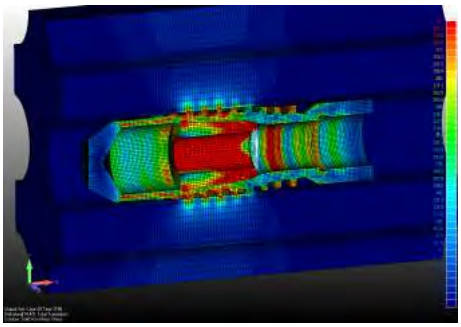

Figure 1.5: Constraint field after the plug insertion.

This mechanical problem involves the three different sources of nonlinearities, which are briefly described next.

Material: we recall that the mechanical behaviour of a material is formulated by a constitutive law relating stress to strain (cf. (1.1d) in the elastic case). There are a lot of possible behaviours and models to describe them. For metallic materials, the constitutive law includes a linear domain, corresponding to a reversible deformation, which is sufficient for a large number of studies. Otherwise, it is necessary to model other behaviours, such as mechanical damage, elastoplasticity or viscoplasticity, which induce irreversible deformations. In such cases, the relation between stress and strain is nonlinear and generally defined by a set of differential equations. In the numerical example introduced above, the tube, the plug and the nut have elastoplastic constitutive laws. More precisely, the nut is considered as perfectly plastic while the tube and the plug obey a strain isotropic hardening rule.

Geometric: some solid mechanics problems are based on a geometric linear approximation considering that displacements and strains are small. Thus, the initial and final positions of a given body are practically the same and the linearized expression of the strain tensor (1.1e) is used.
When the small displacement hypothesis is not satisfied, the compatibility equation is not valid and the Green-Lagrange tensor strain, nonlinear in ∇u, is introduced:

$$e(u) = \frac{1}{2}(\nabla u + \nabla^T u + \nabla^T u \nabla u).$$

Furthermore, the solution of the problem needs to take into account the evolution of the geometry. This is the case in the numerical illustration of this section.
When the small strain hypothesis is not satisfied, the framework of large strains must be used, with the introduction of particular stress and strain measures. In the illustration, a logarithmic strain tensor is used within the framework defined in [6].

Contact: the contact phenomenon is due to the principle of non-penetration of matter: when two bodies are in contact, a force appears on the interaction surface, preventing their penetration, and vanishes when they do not touch anymore. The solution of such a problem leads to the addition of a condition, expressed as an inequation, and the problem turns to be highly nonlinear. Concerning the steam generator tube, contact phenomena occur between the nut and the plug and between the plug and the tube.
It is important to note that this source of nonlinearity is not studied in this present thesis.

1.1.3 Integration within the Newton's method

We aim at presenting the Newton's method applied to nonlinear problems in solid mechanics. More precisely, we just focus here on the example of a nonlinear elastic problem, under the small displacement and strain hypothesis and without any contact condition. Indeed, each type of nonlinearity has to be treated differently, but they all lead to similar sequences of linear systems [21, 35].

Contrary to the linear case introduced in Section 1.1.1, the stress tensor σ does not depend linearly on the strain tensor ϵ any longer. Adapting (1.4) to this case, we search $(u, \lambda) \in \mathcal{C} \times \mathcal{C}'(\Gamma_D)$ such that

$$R(u, \lambda, v, \mu) = \begin{pmatrix} R_1(u, \lambda, v, \mu) \\ R_2(u, \lambda, v, \mu) \end{pmatrix} = 0 \quad \forall (v, \mu) \in \mathcal{C} \times \mathcal{C}'(\Gamma_D),$$

where

$$\begin{cases} R_1(u,\lambda,v,\mu) = \int_\Omega \sigma(u) : \epsilon(v)\ d\Omega + \int_{\Gamma_D} \lambda v\ d\Gamma - \int_\Omega rv\ d\Omega - \int_{\Gamma_N} hv\ d\Gamma \\ R_2(u,\lambda,v,\mu) = \int_{\Gamma_D} (u - u_d)\mu\ d\Gamma \end{cases}.$$

The Newton's method aims at building a sequence (u_i, λ_i) of approximations of the solution (choosing an initial approximation (u_0, λ_0)). At the i-th iteration, the method is based on the first order linearization of R against u_i and λ_i:

$$R(u_i,\lambda_i,v,\mu) \approx R(u_{i-1},\lambda_{i-1},v,\mu) + \nabla_{u,\lambda} R(u_{i-1},\lambda_{i-1},v,\mu) \begin{pmatrix} u_i - u_{i-1} \\ \lambda_i - \lambda_{i-1} \end{pmatrix},$$

with

$$\nabla_{u,\lambda} R(u_{i-1},\lambda_{i-1},v,\mu) = \begin{pmatrix} \frac{\partial R_1(u_{i-1},\lambda_{i-1},v,\mu)}{\partial u} & \frac{\partial R_1(u_{i-1},\lambda_{i-1},v,\mu)}{\partial \lambda} \\ \frac{\partial R_2(u_{i-1},\lambda_{i-1},v,\mu)}{\partial u} & \frac{\partial R_2(u_{i-1},\lambda_{i-1},v,\mu)}{\partial \lambda} \end{pmatrix}.$$

Aiming at vanishing $R(u_i, \lambda_i, v, \mu)$, we search (u_i, λ_i) such that

$$\begin{cases} \nabla_{u,\lambda} R(u_{i-1},\lambda_{i-1},v,\mu) \begin{pmatrix} \delta u_i \\ \delta \lambda_i \end{pmatrix} = -R(u_{i-1},\lambda_{i-1},v,\mu) \\ u_i = u_{i-1} + \delta u_i \\ \lambda_i = \lambda_{i-1} + \delta \lambda_i \end{cases}. \tag{1.6}$$

After using a finite element discretization, and noting that the essential boundary conditions are constant during the linearization process, the Newton's method leads to solve the following sequence of saddle point linear systems:

$$\mathcal{K}_i x_i = b_i \Longleftrightarrow \begin{pmatrix} G_i & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u_i \\ \lambda_i \end{pmatrix} = \begin{pmatrix} f_i \\ g_i \end{pmatrix}, \tag{1.7}$$

where u_i and λ_i now correspond to vectors of size n and m, respectively. Here, the (1,1) blocks G_i are generally symmetric positive semidefinite, as explained in Section 1.1.1, and are supposed to slowly change all along the sequence. In some cases, follower forces occur making these matrices nonsymmetric [5], but the symmetric case is essentially treated in this manuscript (though the nonsymmetric case is covered by the theoretical results in Chapter 3). The saddle point matrices $\mathcal{K}_i$ are then assumed to be symmetric indefinite. As introduced in Section 1.1.1.3, we further note that other linear relations can be added to the discretized problem. We suppose that they involve the same degrees of freedom during the sequence, and that the fixed matrix B can correspond to both single or multifreedom constraints. Finally, we suppose that B is of full rank m and that

the matrices G_i are positive definite on $\mathcal{N}(B)$. Under these assumptions, we show in Section 1.5 the existence and uniqueness of the solution for each system of the sequence (1.7).

Remark 7. The issues related to the convergence of the Newton's method are not discussed here and we refer the reader to [78] for details. Actually, we essentially aimed in this section at introducing the sequence of large-scale saddle point systems that we need to solve.

Remark 8. When contact nonlinearities occur, the matrix B can change during the sequence (1.7), but this case is not treated in this manuscript.

The final purpose of this thesis is to develop efficient methods to solve sequences like (1.7) arising in solid mechanics, especially of large size. With this in mind and considering that the matrices slowly change all along the sequence, we want to use preconditioned Krylov subspace methods [90]. They are detailed in the next section, where we particularly focus on the GMRES method, well suited for the solution of general nonsingular systems, and whose choice will be justified later.

1.2 Krylov subspace methods

1.2.1 Presentation

Let us focus on the solution of the linear system

$$Ax = b, \tag{1.8}$$

where $x, b \in \mathbb{R}^N$ and $A \in \mathbb{R}^{N\times N}$ is a nonsingular sparse matrix. We aim at presenting here the Krylov subspace methods, and particularly the GMRES method which will be used later in this manuscript. The development of these techniques which appeared in the 70's remains an active research domain and we later refer to numerous review papers such as [90], [99] or [110] for more detailed presentations and analysis.

Krylov subspace methods are iterative solvers in the sense that, giving an initial vector x_0, a sequence (x_l) of successive approximations of $x^\star = A^{-1}b$ is built. First, we note that (1.8) is equivalent to the system involving as a right-hand side the initial residual $r_0 = b - Ax_0$ rather than b:

$$Ax = b \Longleftrightarrow A(x - x_0) = r_0.$$

The Cayley-Hamilton theorem proves that the inverse of a matrix can be expressed as a linear combination of its powers. As described in [90], the starting idea of the Krylov subspace methods, at the l-th iteration, is based on the approximation of A^{-1} with a polynomial q_{l-1} of degree at most $l-1$ such that

$$x_l - x_0 = q_{l-1}(A)r_0.$$

Finally, we want to find out $x_l - x_0$ into the subspace spanned by the product of the powers of A and the initial residual.

Definition 1.2.1. *Let $A \in \mathbb{R}^{N\times N}$ a nonsingular matrix, $x_0 \in \mathbb{R}^N$ and $r_0 = b - Ax_0$. The subspace of dimension at most l*

$$\mathcal{K}_l(A, r_0) = span\{r_0, Ar_0, ..., A^{l-1}r_0\}$$

is called the Krylov subspace associated to A and r_0. It is denoted $\mathcal{K}_l$ when there is no ambiguity.

Krylov subspace methods differ particularly from the way we choose x_l into the affine space $x_0 + \mathcal{K}_l$. In its general form, the so called Petrov-Galerkin condition involves the current residual $r_l = b - Ax_l$ and can be written as

$$\text{Find } x_l \in x_0 + \mathcal{K}_l \text{ such as } r_l \perp \mathcal{L}_l, \tag{1.9}$$

where $\mathcal{L}_l$ is some subspace of dimension l and the orthogonality is considered with respect to the canonical inner product.

Before dealing with this condition of orthogonality, we first need to construct a basis of $\mathcal{K}_l$. The natural basis $(r_0, Ar_0, ..., A^{l-1}r_0)$ seems to be the simplest choice but is not interesting from a numerical point of view. Indeed, let us assume that A is diagonalizable with N eigenpairs denoted $\{(u_1, \lambda_1), ..., (u_N, \lambda_N)\}$. The vector $A^j r_0$ can be expressed under the form $A^j r_0 = \sum_{i=1}^{N} \alpha_i \lambda_i^j u_j$ and tends to behave, when j increases, as $\alpha_{max}\lambda_{max}^j u_{max}$ where λ_{max} is the largest eigenvalue in modulus. This choice can lead to a linearly dependence of the basis vectors, that can be troublesome in presence of rounding errors. An orthonormalization process is generally used to avoid this numerical degeneration; several variants can be performed. We present here the Arnoldi method based on the modified Gram-Schmidt process, known to be more stable than the classical variant [43]. This method is detailed in Algorithm 1.

Algorithm 1 Arnoldi method to compute an orthonormal basis of $\mathcal{K}_l(A, r_0)$

choose $x_0 \in \mathbb{R}^N$ and define $r_0 = b - Ax_0$
Define $\beta = \|r_0\|_2$ and $v_1 = r_0/\beta$
for $j = 1, ..., l$ **do**
  $w_j = Av_j$
  for $i = 1, ..., j$ **do**
    $h_{i,j} = w_j^T v_i$
    $w_j = w_j - h_{i,j} v_j$
  end for
  $h_{j+1,j} = \|w_j\|_2$
  $v_{j+1} = w_j / h_{j+1,j}$
end for

Let V_l be defined as $V_l = [v_1, ..., v_l]$ where the column vectors, called Arnoldi vectors, form an orthonormal basis of $\mathcal{K}_l$. The columns of V_l are then orthogonal, as for $V_{l+1} = [V_l, v_{l+1}]$. Furthermore, let us store the orthonormalization coefficients $h_{i,j}$ and define the associated matrix

$$H_{l+1,l} = \begin{pmatrix} h_{1,1} & h_{1,2} & \dots & h_{1,l-1} & h_{1,l} \\ h_{2,1} & h_{2,2} & \dots & h_{2,l-1} & h_{2,l} \\ 0 & h_{3,2} & \ddots & \vdots & \vdots \\ \vdots & \ddots & \ddots & h_{l-1,l-1} & h_{l-1,l} \\ \vdots & & \ddots & h_{l,l-1} & h_{l,l} \\ 0 & \dots & \dots & 0 & h_{l+1,l} \end{pmatrix} \in \mathbb{R}^{l+1,l}.$$

From Algorithm 1 we directly deduce the Arnoldi relation

$$AV_l = V_{l+1} H_{l+1,l}, \tag{1.10}$$

which can be expressed differently, splitting $H_{l+1,l}$ such that

$$H_{l+1,l} = \begin{pmatrix} H_l \\ h_{l+1,l} e_l^T \end{pmatrix}.$$

Here, $H_l \in \mathbb{R}^{l \times l}$ is a square upper Hessenberg matrix and e_l is the l-th column of the identity matrix I_l of order l. Thus

$$AV_l = V_l H_l + h_{l+1,l} v_{l+1} e_l^T \tag{1.11}$$

and

$$V_l^T A V_l = H_l. \tag{1.12}$$

Note that when A is symmetric, H_l is a tridiagonal symmetric matrix and the Arnoldi procedure can be simplified to generate the Lanczos algorithm. The main benefit in this case comes from the basis construction which can be done with a short-term recurrence, just needing v_{i-1} and v_i to compute v_{i+1}. We refer to [99] for details or [65, 66] the historical references. In this manuscript, we use the Lanczos denomination when symmetric matrices are involved, and $H_{l+1,l}$ and $h_{l+1,l}$ are then denoted $T_{l+1,l}$ and $t_{l+1,l}$, respectively.

From now on, we focus on one specific Krylov subspace method, called GMRES [91], which can be applied to general nonsingular linear systems and will be used in this manuscript.

1.2.2 The GMRES method

1.2.2.1 Presentation

The GMRES method is adapted to solve general linear systems as (1.8), respecting the Petrov-Galerkin condition (1.9) with $\mathcal{L}_l = A\mathcal{K}_l$. Thanks to the Arnoldi method described above, we search an approximation of the solution x_l into $x_0 + \mathcal{K}_l$ in the form of

$$x_l = x_0 + V_l y_l,$$

where $y_l \in \mathbb{R}^l$. We know that the column vectors of AV_k form a basis of $\mathcal{L}_l = A\mathcal{K}_l$ and the Petrov-Galerkin condition can be written as

$$V_l^T A^T r_l = 0.$$

From the Arnoldi relation (1.10) and the fact that $v_1 = r_0/\|r_0\|_2$, we obtain

$$\begin{aligned} V_l^T A^T r_l &= V_l^T A^T (b - A x_l), \\ &= V_l^T A^T r_0 - V_l^T A^T A V_l y_l, \\ &= \|r_0\|_2 H_{l+1,l}^T V_{l+1}^T v_1 - H_{l+1,l}^T H_{l+1,l} y_l, \\ &= H_{l+1,l}^T (\beta e_1 - H_{l+1,l} y_l), \end{aligned}$$

where $\beta = \|r_0\|_2$ and $e_1 = [1, 0, ..., 0]^T \in \mathbb{R}^l$. Finally, we need to solve the equation

$$H_{l+1,l}^T(\beta e_1 - H_{l+1,l} y_l) = 0$$

which corresponds to the normal equation of the minimization problem

$$y_l = \underset{y \in \mathbb{R}^l}{\operatorname{argmin}} \|\beta e_1 - H_{l+1,l} y\|_2. \tag{1.13}$$

To summarize, computing y_l is equivalent to solve a least-squares problem of size $(l+1)\times l$ and the approximate solution x_l during the l-th iteration of GMRES is given by $x_l = x_0 + V_l y_l$ (see Algorithm 2).

Algorithm 2 Main steps of GMRES to solve $Ax = b$

choose $x_0 \in \mathbb{R}^N$ and define $r_0 = b - Ax_0$
Define $\beta = \|r_0\|_2$ and $v_1 = r_0/\beta$
 # Arnoldi process
Compute V_{l+1} and $H_{l+1,l}$ such that $AV_l = V_{l+1}H_{l+1,l}$
 # Least-squares problem
$y_l = \underset{y \in \mathbb{R}^l}{\operatorname{argmin}} \|\beta e_1 - H_{l+1,l} y\|_2$
 # Approximation of the solution
$x_l = x_0 + V_l y_l$

Before discussing how to solve the minimization problem (1.13), we prove in Proposition 1.2.1 that GMRES can be presented as a minimum residual problem hence its name, standing for "General Minimum RESidual".

Proposition 1.2.1. *To satisfy the Petrov-Galerkin condition* (1.9) *with* $\mathcal{L}_l = A\mathcal{K}_l$ *amounts to solve the minimization problem*

$$\|r_l\|_2 = \min_{y \in \mathbb{R}^l} \|\beta e_1 - H_{l+1,l} y\|_2 = \min_{x \in x_0 + \mathcal{K}_l} \|Ax - b\|_2.$$

Proof. Let y_l be the solution of (1.13) and $x_l = x_0 + V_l y_l$ which satisfies the Petrov-Galerkin condition. Since $V_{l+1}^T V_{l+1} = I_{l+1}$,

$$\begin{aligned} \|r_l\|_2 &= \|r_0 - AV_l y_l\|_2, \\ &= \|V_{l+1}(\beta e_1 - H_{l+1,l} y_l)\|_2, \\ &= \|\beta e_1 - H_{l+1,l} y_l\|_2. \end{aligned}$$

The fact that y_l is the solution of (1.13) completes the proof. □

1.2.2.2 Least-squares problem

To complete the presentation of the GMRES method, we need to detail how to solve (1.13). In practice, a QR factorization of $H_{l+1,l}$ is usually performed, using Givens rotation matrices of the form

$$G_i = \begin{pmatrix} I_{i-1} & 0 & 0 \\ 0 & \begin{pmatrix} c_i & s_i \\ -s_i & c_i \end{pmatrix} & 0 \\ 0 & 0 & I_{l-i} \end{pmatrix} \in \mathbb{R}^{(l+1)\times(l+1)},$$

where $c_i = cos(\theta_i)$ and $s_i = sin(\theta_i)$ are chosen in order to eliminate the subdiagonal element $h_{i+1,i}$ of $H_{l+1,l}$. Computing these matrices recursively and denoting their product by $Q_l = G_{l+1}G_l...G_1$, we obtain

$$Q_l H_{l+1,l} = \begin{pmatrix} R_l \\ 0 \end{pmatrix},$$

where $R_l \in \mathbb{R}^{l\times l}$ is an upper triangular matrix. Q_l is obviously orthogonal and the minimization problem (1.13) can be replaced by

$$y_l = \underset{y\in\mathbb{R}^l}{\operatorname{argmin}} \|\beta Q_l e_1 - \begin{pmatrix} R_l \\ 0 \end{pmatrix} y\|_2.$$

Denoting $\beta Q_l e_1 = [u_1, ..., u_l, u_{l+1}]^T$, the expected solution is then $y_l = R_l^{-1}[u_1, ...u_l]^T$ which can easily be performed by backward substitution. We further note that Proposition 1.2.1 leads to the fact that the current residual verifies $\|r_l\|_2 = |u_{l+1}|$. This relation is particularly useful to check if the norm of the residual is small enough to stop the GMRES process and obtain the expected solution of the linear system $Ax = b$. More specifically, the stopping criterion is often evaluated using the relative residual $\|r_l\|_2/\|r_0\|_2$, as one can see in the implementation of GMRES described in Algorithm 3. It is interesting to note that this iterative process can be implemented such that each iteration requires only one matrix-vector product of the form Ay.

Algorithm 3 GMRES algorithm to solve $Ax = b$

Choose a convergence threshold ϵ
Choose $x_0 \in \mathbb{R}^N$ and define $r_0 = b - Ax_0$
Define $\beta = \|r_0\|_2$, $v_1 = r_0/\beta$ and $u_1 = \beta e_1$.
for $i = 1, ...$ **do**
  # Arnoldi process
  $w_i = Av_i$
  for $j = 1, ..., i$ **do**
    $h_{j,i} = w_i^T v_j$
    $w_i = w_i - h_{j,i} v_i$
  end for
  $h_{i+1,i} = \|w_i\|_2$
  $v_{i+1} = w_i / h_{i+1,i}$
  # Least-squares problem
  for $j = 2, ..., i$ **do**
    $r_{j-1,i} = c_{j-1} h_{j-1,i} + s_{j-1} h_{j,i}$
  end for
  $\gamma = \sqrt{h_{i,i}^2 + h_{i+1,i}^2}$
  $c_i = h_{i,i}/\gamma$, $h_{i+1,i}/\gamma$, $r_{i,i} = c_i h_{i,i} + s_i h_{i+1,i}$
  $u_{i+1} = -s_i u_i$, $u_i = c_i u_i$
  if $|u_{i+1}| \leq \epsilon\beta$ **then**
    $I = i$
    end for i
  end if
end for
$y_I = R_I^{-1}[u_1, ..., u_I]^T$
 # Approximation of the solution
$x = x_0 + V_I y_I$

1.2.2.3 Convergence analysis

We start this convergence analysis of GMRES with the fact that the norm of the residual decreases along the convergence. In fact, we note that the successive Krylov subspaces are embedded and Proposition 1.2.1 leads to

$$\|r_{l-1}\|_2 \geq \|r_l\|_2.$$

If $l < N$ exists such that $\mathcal{K}_l(A, r_0) = \mathcal{K}_{l+1}(A, r_0)$, the equality between x_l and the solution $x^\star = A^{-1}b$, in exact real arithmetic, for A nonsingular can be proved (see, e.g., [50]). Finally, the process is stopped. If such a case does not happen, we have finally $\mathcal{K}_N(A, r_0) = \mathbb{R}^N$ and x_N is obviously the expected solution.

Moreover, we have seen at the beginning of this current section that the approximate solution x_l is searched as $x_l = x_0 + q_{l-1}(A)r_0$, where q_{l-1} is a polynomial of degree at most $l-1$. The associated residual can be written as

$$\begin{aligned}\|r_l\|_2 &= \|b - Ax_l\|_2, \\ &= \|(I_N - Aq_{l-1}(A))r_0\|_2.\end{aligned}$$

Then we obtain a bound for the residual of the l-th iteration given by

$$\|r_l\|_2 \leq \|r_0\|_2 \min_{p \in \mathcal{P}_l} \|p(A)\|_2$$

where

$$\mathcal{P}_l = \{p \mid p \text{ is a polynomial of degree at most } l \text{ with } p(0) = 1\}.$$

To conclude, we present a more accurate result in the case of A being diagonalizable, proved for instance in [91].

Theorem 1.2.2. *Let $A \in \mathbb{R}^{N\times N}$ be nonsingular and diagonalizable with the spectral decomposition $A = U\Lambda U^{-1}$. The GMRES method produces at the l-th iteration a residual which satisfies the inequality*

$$\|r_l\|_2 \leq \|r_0\|_2 \kappa(U) \min_{p \in \mathcal{P}_l} \max_{i=1,\ldots,N} |p(\lambda_i)|$$

where $\kappa(U) = \|U\|_2\|U^{-1}\|_2$ is the condition number of U and $\Lambda = diag(\lambda_1, ..., \lambda_N)$.

This bound on the relative residual is instructive in the sense that the convergence of GMRES is influenced by the minimization of a polynomial over the set of eigenvalues of A. More precisely, if U is not too ill-conditioned, this bound mainly depends on the

spectrum of the matrix. When A is symmetric, the similarity transformation matrix U is orthogonal, i.e. $\kappa(U) = 1$. It is more complicated in the nonsymmetric case, and the spectrum alone is not sufficient to describe the convergence behavior of GMRES [47]. Nevertheless, the existence of clustered eigenvalues can speedup the solution (see e.g. [24, 97, 14]). We also refer the reader to the recent review paper [69] detailing the role of the spectrum in forming GMRES residual norms.

1.2.2.4 Restarted variant

A large number of iterations may be necessary to obtain an acceptable solution with GMRES, particularly when solving large-scale linear systems. This can be problematic in terms of memory requirement, since one additional vector of size N has to be stored at each iteration (a basis vector of the Krylov subspace). Moreover, the dimension of the least-squares problem to solve after the Arnoldi orthogonalization increases with the number of iterations. To remedy this situation, GMRES is usually restarted after each m steps defining the variant called GMRES(m) [91]. This method performs cycles of m iterations, restarting with a vector x_0 equal to the last computed iterate x_m. The dimension of the Krylov subspace $\mathcal{K}_m$ is then controlled and the memory requirements are limited. The value of this restart parameter will be fixed at 30 in the different numerical experiments of this thesis.

1.2.3 Ritz and harmonic Ritz pairs

As mentioned before, the convergence rate of the Krylov subspace methods can be related to the spectrum of the matrix A. In this sense, we detail two ways to deduce approximated eigenpairs of A, from the Arnoldi method, thanks to Ritz or harmonic Ritz pairs. This presentation is based on different review papers (see, e.g., [100], [82], [73] or Section 4.3 in [7]). Both approximating strategies can be introduced as solution of the problem

$$\begin{cases} w \in \mathcal{K}_l \\ Aw - \theta w \perp \mathcal{L}_l \end{cases}, \tag{1.14}$$

where $\mathcal{K}_l$ is the currently available Krylov subspace, $\mathcal{L}_l$ is a given subspace of dimension l and the orthogonality is considered with respect to the canonical inner product.

1.2.3.1 Ritz pairs

First, we consider the case where $\mathcal{L}_l = \mathcal{K}_l$: using the notation coming from the Arnoldi method described in Section 1.2.1, the conditions (1.14) lead to the solution of the

equation

$$V_l^T(AV_l y - \theta V_l y) = 0 \quad \text{where} \quad w = V_l y.$$

Using (1.12) and the orthogonality of the columns of V_l, we need to solve $H_l y - \theta y = 0$, i.e. to find the eigenpairs of the matrix H_l of order l. Besides, we recall that in a GMRES(m) context, l is less than m and this eigenproblem can be solved cheaply.

Definition 1.2.2. *Let A be a nonsingular matrix and assume that l iterations of the Arnoldi method have been performed so that the Arnoldi relation* (1.12) *holds. A Ritz pair is defined as a pair $(w = V_l y, \theta) \in \mathbb{C}^N \times \mathbb{C}$ where $(y, \theta) \in \mathbb{C}^l \times \mathbb{C}$ is an eigenpair of H_l. w is called the Ritz vector associated to the Ritz value θ.*

A Ritz vector is built from an eigenvector of H_l which represents the compression of A onto the Krylov subspace $\mathcal{K}_l$. We emphasize that if A is symmetric, H_l is also symmetric and the Ritz pairs are real-valued. In the more general case of A being real-valued and nonsingular, these pairs can be complex-valued. To conclude, we can easily express the error coming from this spectral approximation using the Arnoldi relation (1.11):

$$\begin{aligned} Aw &= AV_l y, \\ Aw &= V_l H_l y + h_{l+1,l} v_{l+1} e_l^T y, \\ Aw &= \theta w + (e_l^T y) h_{l+1,l} v_{l+1}. \end{aligned}$$

Since $\|v_{l+1}\|_2 = 1$, this error can finally be written as

$$\|Aw - \theta w\|_2 = |(e_l^T y) h_{l+1,l}|. \tag{1.15}$$

1.2.3.2 Harmonic Ritz pairs

Another interesting case arises when $\mathcal{L}_l = A\mathcal{K}_l$. Taking AV_l as a basis of $\mathcal{L}_l$, the orthogonality relation (1.14) becomes

$$V_l^T A^T(AV_l y - \theta V_l y) = 0 \quad \text{where} \quad w = V_l y.$$

The equality (1.10) leads to

$$\begin{aligned} V_l^T A^T A V_l y &= H_{l+1,l}^T H_{l+1,l} y, \\ &= \begin{pmatrix} H_l \\ h_{l+1,l} e_l^T \end{pmatrix}^T \begin{pmatrix} H_l \\ h_{l+1,l} e_l^T \end{pmatrix} y, \\ &= H_l^T H_l y + |h_{l+1,l}|^2 e_l e_l^T y, \end{aligned}$$

and

$$\begin{aligned} V_l^T A^T V_l y &= H_{l+1,l}^T V_{l+1}^T V_{l+1} \begin{pmatrix} y \\ 0 \end{pmatrix}, \\ &= H_{l+1,l}^T \begin{pmatrix} y \\ 0 \end{pmatrix}, \\ &= H_l^T y. \end{aligned}$$

Eventually, we need to solve the eigenproblem

$$(H_l + |h_{l+1,l}|^2 H_l^{-T} e_l e_l^T) y = \theta y.$$

Definition 1.2.3. *Let A be a nonsingular matrix and assume that l iterations of the Arnoldi method have been performed so that the Arnoldi relation* (1.12) *holds. An harmonic Ritz pair is defined as a pair $(w = V_l y, \theta) \in \mathbb{C}^N \times \mathbb{C}$ where $(y, \theta) \in \mathbb{C}^l \times \mathbb{C}$ is an eigenpair of $H_l + |h_{l+1,l}|^2 H_l^{-T} e_l e_l^T$. We call w the harmonic Ritz vector associated to the harmonic Ritz value θ.*

As remarked before, the dimension of the eigenproblem is moderate when using GMRES(m); these spectral approximations are thus easily available. However, contrary to the Ritz case, no simple characterization of the approximation error is known. The Harmonic Ritz pairs seem to be naturally associated to GMRES given that the same projection subspace $\mathcal{L}_l = A\mathcal{K}_l$ is used. Besides, it can be shown that the Harmonic Ritz values are the zeros of the residual polynomial $p_l \in \mathcal{P}_l$ which satisfies $r_l = p_l(A) r_0$ [44]. Nevertheless, it is common in practice to use Ritz or harmonic Ritz pairs as approximate eigenpairs with this Krylov subspace method. Both pairs are related, since the harmonic Ritz values can be seen as inverses of Ritz values for A^{-1}, although with respect to a subspace that is generated for A [100]. Lastly, the Ritz ones often tend to provide better approximations for extremal eigenvalues, and the harmonic Ritz values can be expected to be a better choice to approximate interior eigenvalues [72, 73, 57].

1.2.4 Preconditioning

As mentioned in [90], Krylov subspace methods are only feasible in combination with the concept of preconditioning when considering large-scale problems. This notion is central in this thesis and refers to transforming the original system $Ax = b$ in a new one, maintaining the same solution but getting more favorable properties for the convergence of iterative methods. The rate of convergence is a complex issue, especially when nonsymmetric matrices are involved, but it is known that the behaviour is notably linked to the spectral distribution (see the discussion in Section 1.2.2.3). Precisely, a fast convergence can be obtained when clusters of eigenvalues away from 0 appear [24]. From this perspective, we can transform the original system in different ways, using a nonsingular matrix M:

- $MAx = Mb$ (*left preconditioning*)
- $\begin{cases} AM\hat{x} = b \\ x = M\hat{x} \end{cases}$ (*right preconditioning*)
- $\begin{cases} M_1AM_2\hat{x} = M_1b \\ x = M_2\hat{x} \end{cases}$ with $M = M_1M_2$ (*split preconditioning*)

The choice of M is subject to some compromises: the application of M or M_1 and M_2 on a vector must not be expensive and the preconditioned matrix MA (respectively AM or M_1AM_2) is expected to have favorable properties to decrease the number of iterations of the Krylov subspace method. In fact, the preconditioner is either the inverse of an approximation of A (forward type) or an approximation of the inverse of A (inverse type). It is worth mentioning that the number of iterations of the Krylov subspace method is generally different if M is used as a left, right or split preconditioner, even though the spectra of the associated preconditioned matrices are identical. In particular, the stopping criterion is evaluated using the norm of the relative preconditioned residual which is different for the three strategies. Besides, right preconditioning can be attractive, since the preconditioned residual is equal to the original one.

Designing preconditioners is an active domain of research (see e.g. [11, 110]) and there are generally two approaches to construct such techniques. The first one is a physics-based approach, which requires a complete knowledge of the underlying problem. We can cite the geometric multigrid preconditioners, whose typical application is in the numerical solution of elliptic partial differential equations on structured grids. We refer the reader to [108] for more details. The second approach is related to algebraic methods

which only use information contained in the coefficients of the matrix A. We succinctly describe three popular algebraic preconditioning techniques:

- **Incomplete LU factorization** (Chapter 10 in [90]): this technique is based on the Gaussian elimination that provides the factorization $A = LU$ where L and U are respectively lower and upper triangular matrices. The limit of this method, which can be used as a direct method to solve $Ax = b$, concerns the fill-in character of this factorization: even if A is sparse, the factors L and U can be much more dense than A. In solid mechanics, this phenomena usually occurs when a 3D problem is studied. To remedy this situation, renumbering can be performed. The main idea of the relied preconditioning techniques is to control the fill-in phenomena defining P as a set of couples $(i, j) \in \{1, ..., N\}^2$ and computing the $\tilde{L}\tilde{U}$ factorization under the conditions

$$\tilde{L}_{i,j} = 0 \text{ for } (i, j) \in P \text{ and } i > j$$

$$\tilde{U}_{i,j} = 0 \text{ for } (i, j) \in P \text{ and } i \leq j.$$

  The particular case where P corresponds to the location of the zeros of A is referred as $ILU(0)$. It is possible to generalize this procedure calling $ILU(k)$ the incomplete LU factorization when P defines the sparsity pattern of the matrix A^{k+1}. The associated preconditioner $M = (\tilde{L}\tilde{U})^{-1}$ is finally applied thanks to successive forward and backward substitutions. We remark that for A symmetric positive definite, this technique can be adapted to perform an incomplete Cholesky factorization (denoted $ICC(k)$).

- **Sparse approximate inverses** [16]: the idea is to define a sparse matrix M which approximates A^{-1} under the condition

$$M = \underset{M \in \mathcal{S}}{\operatorname{argmin}} \|I_N - AM\|_F,$$

  where $\mathcal{S}$ is a sparsity pattern to impose and $\|.\|_F$ is the Frobenius norm. One can show that the minimization of this norm can be split in N independent least-squares problems to compute each column of M. Several choices exist for $\mathcal{S}$ but we refer to [49] for the most successful method, called SPAI, which uses an adaptive strategy for the selection of the sparsity pattern. This preconditioning technique can be very powerful but often very expensive in terms of computational time, even in a parallel computing framework.

- **Algebraic Multigrid (AMG)** [103]: contrary to the physics-based multigrid approach, no geometrical background on the problem is necessary and this technique can be applied to discretizations on unstructured grids.

Note that the preconditioners used and developed in this thesis all belong to the algebraic approach. To conclude, we detail the preconditioned GMRES(m) method in Algorithm 4. For a generic description, we use a split preconditioner, the left case being obtained for $M_2 = I_N$ and the right one for $M_1 = I_N$. In a few words, we search the approximation of the solution x_l into $x_0 + M_2\mathcal{K}_l(M_1 r_0, M_1 A M_2 r_0)$, minimizing the norm of the residual $\|r_l\|_2 = \|M_1(b - Ax_l)\|_2$ on this affine subspace. We note that each iteration of GMRES requires one application of A, M_1 and M_2.

Remark 9. In this manuscript, we refer to $M_1(b - Ax_l)$ and $b - Ax_l$ as the preconditioned and actual residuals, respectively.

Algorithm 4 GMRES(m) algorithm with a split preconditioner

Choose a convergence threshold ϵ
Choose $x_0 \in \mathbb{R}^N$ and define $r_0 = M_1(b - Ax_0)$
Define $\beta = \|r_0\|_2$, $v_1 = r_0/\beta$ and $u_1 = \beta e_1$.
for $l = 1, ...$ **do**
  for $i = 1, ..., m$ **do**
    $w_i = M_1 A M_2 v_i$
    for $j = 1, ..., i$ **do**
      $h_{j,i} = w_i^T v_j$
      $w_i = w_i - h_{j,i} v_i$
    end for
    $h_{i+1,i} = \|w_i\|_2$
    $v_{i+1} = w_i / h_{i+1,i}$
  end for
  $y_m = \underset{y \in \mathbb{R}^m}{\operatorname{argmin}} \|\beta e_1 - H_{m+1,m} y\|_2$
  if $|u_{m+1}| \leq \epsilon \times \beta$ (where u_{m+1} is defined in Section 1.2.2.2) **then**
    $x = x_0 + M_2 V_m y_m$
  end if
  $x_0 = x_0 + M_2 V_m y_m$
  $r_0 = M_1(b - Ax_0)$
end for

We have detail how to solve a linear system using the Krylov subspace method called GMRES. It is now important to take into account the saddle point structure of the linear

systems of (1.7). That is the main purpose of Section 1.3 which notably reviews existing preconditioning techniques for such systems.

1.3 Linear systems of saddle point structure

The saddle point structure arises in many applications in science and engineering, including for instance constrained optimization [30], finite element discretization in fluid dynamics [32] or in electromagnetism [86]. Hence, there has been in recent years a growth of interest in saddle point problems, and many solution techniques have been developed. We refer the reader to [14] for a large survey. Throughout the remainder of the manuscript, and according to Section 1.1, we consider the linear system

$$\mathcal{K}x = b \Longleftrightarrow \begin{pmatrix} G & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix}, \tag{1.16}$$

where $G \in \mathbb{R}^{n\times n}$ is symmetric positive semidefinite, $B \in \mathbb{R}^{m\times n}$, $f \in \mathbb{R}^n$, $g \in \mathbb{R}^m$ and $m \leq n$. This symmetric indefinite problem of dimension $N = n+m$ is assumed to belong to the class of large and sparse saddle point problems. The saddle point system (1.16) can also be written as

$$\begin{cases} Gu + B^T\lambda = f \\ Bu = g \end{cases}. \tag{1.17}$$

Before presenting solution methods for (1.16), we state a theorem related to the nonsingularity of $\mathcal{K}$ which makes sure the existence and uniqueness of the solution. Theorem 1.3.1 is adapted from Lemma 1.1 in [13], and the related assumptions will be supposed to be satisfied from now on. This result can be particularly applied on each linear system of (1.7).

Theorem 1.3.1. *Let $\mathcal{K} \in \mathbb{R}^{N\times N}$ be the coefficient matrix in* (1.16)*. Assume that G is symmetric positive semidefinite, B has full rank m and $\mathcal{N}(G) \cap \mathcal{N}(B) = \{0\}$. Then $\mathcal{K}$ is nonsingular.*

Proof. We focus here on the null space of $\mathcal{K}$ solving the equation $\mathcal{K}x = 0$, or equivalently

$$\begin{cases} Gu + B^T\lambda = 0 \\ Bu = 0 \end{cases}, \tag{1.18}$$

which implies directly that $u \in \mathcal{N}(B)$. Using this statement and premultiplying the first

equality in (1.18) by u^T, we obtain

$$u^T G u + u^T B^T \lambda = 0$$

and finally

$$u^T G u = 0.$$

The symmetric semidefinite positiveness of G implies that $u \in \mathcal{N}(G)$ and, from the assumption $\mathcal{N}(G) \cap \mathcal{N}(B) = \{0\}$, we conclude that $u = 0$. Then, replacing u by 0 in the first equality of (1.18), the full rank property of B leads to the nullity of λ. The solution of $\mathcal{K}x = 0$ is thus trivial and $\mathcal{K}$ is nonsingular. □

In this manuscript, we focus on the case where the (1,1) block G is symmetric positive semidefinite. As we will see later in this section, this property can imply some restrictions about the choice of the method used to solve the saddle point system (1.16). Theory has been developed in this sense, called the augmented-Lagrangian approach (see e.g. [41, 42, 81]), which is based on the transformation of the original system (1.16) into the equivalent one

$$\begin{pmatrix} G + B^T W B & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} f + B^T W g \\ g \end{pmatrix}. \tag{1.19}$$

The matrix $W \in \mathbb{R}^{m \times m}$ has to be suitably determined. Choosing W as a symmetric positive definite matrix seems to be advantageous since this property added to the fact that $\mathcal{N}(G) \cap \mathcal{N}(B) = \{0\}$ implies the symmetric positive definiteness of $G + B^T W B$. In particular, this block is nonsingular. Hence, we can consider that the (1,1) block is nonsingular, without loss of generality. We aim at exposing in this part different methods to solve (1.17), which can be split into two families: the "segregated" and the "coupled" methods.

1.3.1 Segregated solvers

This family consists in computing successively u and λ in (1.16). We present here the Schur complement reduction and the null space methods.

1.3.1.1 Schur complement reduction

Let us consider that G is nonsingular, which can be made possible with the augmented-Lagrangian approach introduced above. The idea is to multiply the first equation in (1.17) by BG^{-1} and to subtract the second relation to deduce the equation satisfied by

λ:

$$BG^{-1}B^T\lambda = BG^{-1}f - g. \tag{1.20}$$

The matrix $S = -BG^{-1}B^T$ is known as the Schur complement of the saddle point system. Once the solution $\lambda^\star$ of (1.20) has been computed, u finally satisfies the equation

$$Gu = f - B^T\lambda^\star.$$

The efficiency of this method is particularly related to the properties of the matrices G and B. Indeed, the Schur complement matrix is generally dense, especially when B contains one or more dense rows which leads to a fill-in phenomena (see e.g. Chapter 13.2 in [90]). Thus, the situation is favorable when the linear system of the form $Gy = z$ is easy to solve, for instance when a factorization of this matrix can be cheaply computed. In this case, S may not be formed explicitly and it is possible to use an iterative method to solve (1.20), which only needs matrix-vector products of the form Sy.

1.3.1.2 Null space methods

On the other hand, the null space method involves the knowledge of a basis of $\mathcal{N}(B)$ [14, 26]. Hence, denoting Z a matrix such that $BZ = 0$, the procedure can be summarized as follows:

- Knowing $\hat{u}$ as a particular solution of $Bu = g$, we can search u in the form of $u = \hat{u} + Zw$. If we substitute this expression in the first equality of (1.17) and we premultiply the result by Z^T, w is the solution of

  $$Z^TGZw = Z^T(f - G\hat{u}).$$

  The matrix on the left-hand side is of order $n - m$ and the assumption $\mathcal{N}(G) \cap \mathcal{N}(B) = \{0\}$ ensures its nonsingularity. We denote the solution by $w^\star$ and $u^\star = \hat{u} + Zw^\star$.

- Using the second equation of (1.17), λ is finally the solution of the overdetermined system

  $$B^T\lambda = f - Gu^\star. \tag{1.21}$$

Several issues need to be handled, especially for the computation of the matrix Z, the vector $\hat{u}$ and the solution of (1.21). An overview about these issues is described in [14]. We emphasize that this method does not involve the matrix G^{-1} and is adapted when

G is singular, as long as the assumption $\mathcal{N}(G) \cap \mathcal{N}(B) = \{0\}$ is satisfied. Furthermore, it can be attractive when a sequence of saddle point linear systems needs to be solved with a fixed matrix B and a large number of constraints m, since the associated matrix Z can be computed only once. We remark that this approach is currently investigated in Code_Aster and we refer the reader to [26] for more details. However, the numerical results are not totally satisfactory, particularly in terms of performance and robustness issues. Hence, the methods developed in this manuscript related to the improvement of preconditioning techniques are relevant for the problems studied at EDF. They belong to the class of coupled solvers described next.

1.3.2 Coupled solvers

The coupled methods are different from the segregated ones, in the sense that they compute u and λ simultaneously, using the whole system (1.16).

1.3.2.1 Global solvers (with description of the available solvers in Code_Aster)

Obviously, it is possible to treat the global matrix of (1.16) without taking into account the saddle point structure. On the one hand, the direct solvers can be used, based on sparse Gaussian factorization of the matrix $\mathcal{K}$. We present here two methods available in Code_Aster [19]:

- *MULT_FRONT* is an inhouse multifrontal method, parallelized in shared memory (OpenMP). This method does not use any pivoting, and a breakdown can occur due to zeros on the diagonal of the $(2,2)$ block in (1.16). To overcome this situation, the original system is transformed into an equivalent one, using the notion of "double Lagrange" multipliers [85]:

$$\begin{pmatrix} G & \alpha B^T & \alpha B^T \\ \alpha B & -\alpha I_m & \alpha I_m \\ \alpha B & \alpha I_m & -\alpha I_m \end{pmatrix} \begin{pmatrix} u \\ \lambda_1 \\ \lambda_2 \end{pmatrix} = \begin{pmatrix} f \\ \alpha g \\ \alpha g \end{pmatrix}, \text{ with } \begin{cases} \lambda_1 = \lambda_2 \\ \lambda = \alpha(\lambda_1 + \lambda_2) \end{cases}. \tag{1.22}$$

  We note that $Bu = g$ is multiplied by an arbitrary coefficient $\alpha > 0$, chosen in order to obtain coefficients in the matrix αB of similar magnitude with the ones of G. In practice, as described in [85], α is equal to the average of the extremal values of the diagonal terms of G. This coefficient will be used later in this manuscript.

- Since a few years, it is possible to factorize the matrix of (1.16) using the MUMPS library [2]. The methods from this software allow the treatment of symmetric

indefinite matrices (LDL^T factorizations) in a distributed memory environment.

On the other hand, (1.16) can be solved with preconditioned Krylov subspace methods. Different techniques are accessible in Code_Aster, such as:

- An inhouse preconditioned conjugate gradient method (PCG), used for both symmetric positive definite and indefinite linear systems. Two types of preconditioners can be used, including an incomplete LU factorization ($LDLT\_INC$), or a complete direct factorization performed in single precision arithmetics using the MUMPS library ($LDLT\_SP$).
- Several Krylov subspace methods such as GMRES, performed with the PETSc library [3]. Both preconditioning factorizations $LDLT\_INC$ and $LDLT\_SP$ can be computed, as well as different algebraic multigrid preconditioners (see [19] for more details).

In practice, the relevance of MUMPS, either used as a direct solver or as a preconditioner via $LDLT\_SP$, has been shown in Code_Aster. In general, the direct solver variant is used to solve one or several linear systems with a fixed left-hand side. Actually, the factorization of this matrix is computed once and reused to solve the possible other linear systems. When a sequence with slowly varying matrices is treated, the preconditioning technique $LDLT\_SP$ combined with a Krylov subspace method is often the method of choice: the factorization in single precision arithmetics of the first matrix can also been used as a preconditioner for the subsequent ones. Precisely, when the effect of this preconditioner declines during the sequence, a new factorization can be performed. Using single precision arithmetics for the preconditioner is interesting in the sense that we can roughly divide by two the memory consumption and simultaneously decrease the computational time of the factorization. Finally, we note that even for the solution of one linear system, a preconditioned Krylov subspace method can be more efficient than a direct solver. In fact, a fill-in phenomenon, already described in Section 1.2.4, can occur during the factorization, and the $LDLT\_SP$ preconditioner combined with an iterative method can be more attractive, especially in terms of memory requirements.

In this thesis, we propose preconditioners update formulas to improve the convergence rate of Krylov subspace methods applied to (1.16). In particular, we will see in Chapter 3 that one of these techniques can improve the effect of the $LDLT\_SP$ preconditioner. Alternatively, we provide next an overview of block preconditioners for saddle point systems existing in the literature, which can give other attractive candidates.

1.3.2.2 Block preconditioners for saddle point systems

This issue is widely studied in the literature, with specific applications. For instance, we refer the reader to [14] for a large overview, to [96] for generalized saddle point systems with application to Navier-Stokes problems, or to the PhD thesis [61]. We first recall the type of saddle point matrix of interest:

$$\mathcal{K} = \begin{pmatrix} G & B^T \\ B & 0 \end{pmatrix},$$

where G is real-valued and symmetric positive semidefinite. The case where G is symmetric positive definite in the whole space $\mathbb{R}^N$ has been largely analysed by many authors [15, 17, 98, 79], and we can possibly use the augmented-Lagrangian technique to be compatible with this framework. We aim at presenting here some popular block preconditioners, split into several classes, without being exhaustive. Note that they involve the effect of nonsingular matrices G_0 and S_0 which approximate G and the Schur complement $S = -BG^{-1}B^T$ respectively. In the following, matrices $\mathcal{M}$ are approximations of $\mathcal{K}$ and we use $\mathcal{M}^{-1}$ as a preconditioner for Krylov subspace methods.

- **Block diagonal preconditioners**
  The matrix
  $$\mathcal{M} = \begin{pmatrix} G_0 & 0 \\ 0 & -S_0 \end{pmatrix}$$
  is a block diagonal preconditioner. In [74], the authors prove that for $G_0 = G$ and $S_0 = S$, the spectrum of $\mathcal{M}^{-1}\mathcal{K}$ is reduced to the 3 eigenvalues $\{1, \frac{1}{2} \pm \frac{\sqrt{5}}{2}\}$. This result is theoretically attractive, since the associated preconditioned Krylov subspace method will terminate in at most 3 iterations. Nevertheless, this exact preconditioner can not be used in practice, since its use is approximately as expensive as computing the inverse of the saddle point matrix $\mathcal{K}$ [14]. In this sense, an analysis of the eigenvalue distribution is made in [96], particularly when $G_0 \simeq G$ and $S_0 = -BG_0^{-1}B^T$.

- **Block triangular preconditioners**
  Let us introduce the upper triangular preconditioner
  $$\mathcal{M} = \begin{pmatrix} G_0 & B^T \\ 0 & S_0 \end{pmatrix}.$$

The application of $\mathcal{M}^{-1}$ as a preconditioner can be based on the factorization

$$\mathcal{M} = \begin{pmatrix} G_0 & 0 \\ 0 & S_0 \end{pmatrix} \begin{pmatrix} I_n & G_0^{-1}B^T \\ 0 & I_m \end{pmatrix}.$$

For $G_0 = G$ and $S_0 = S$, a direct calculation leads to

$$\mathcal{K}\mathcal{M}^{-1} = \begin{pmatrix} I_n & 0 \\ BG^{-1} & I_m \end{pmatrix},$$

then $\Lambda(\mathcal{M}^{-1}\mathcal{K}) = \{1\}$. As mentioned before, approximations of G and S have to be used in practice and we especially refer to [62] for a spectral characterization. The following block lower triangular preconditioner can also be used with similar properties:

$$\mathcal{M} = \begin{pmatrix} G_0 & 0 \\ B & S_0 \end{pmatrix}.$$

- **Constraint preconditioners**
  The matrix
  $$\mathcal{M} = \begin{pmatrix} G_0 & B^T \\ B & 0 \end{pmatrix}$$
  is known as a constraint preconditioner. In [58] and [28], a precise spectral analysis is produced. We also refer to the study about inexact constraint preconditioners in an interior point framework, in [17]. The idea is to use an approximation $B_0 \simeq B$ and to characterize the spectrum of the preconditioned matrix.
  The application of $\mathcal{M}^{-1}$ as a preconditioner can be based on the decomposition
  $$\mathcal{M} = \begin{pmatrix} I_n & 0 \\ BG_0^{-1} & I_m \end{pmatrix} \begin{pmatrix} G_0 & 0 \\ 0 & S_0 \end{pmatrix} \begin{pmatrix} I_n & G_0^{-1}B^T \\ 0 & I_m \end{pmatrix},$$
  where $S_0 = -BG_0^{-1}B^T$.

- **Saddle point and nonstandard inner products**
  One can note that most of the block preconditioners introduced above are either nonsymmetric or symmetric indefinite and the GMRES method described in Section 1.2 can be used as the Krylov subspace method. However, some authors have considered finding a nonstandard inner product, defined by a block diagonal matrix, in which the preconditioned saddle point matrix is self-adjoint [22, 102, 87, 64]. Hence, e.g. the MINRES and Conjugate Gradient methods, which are adapted to

solve respectively symmetric indefinite and symmetric positive definite systems, can be used with this nonstandard inner product.

All these preconditioning techniques seem to be attractive but the choice of $G_0 \simeq G$ and especially the choice of $S_0 \simeq S$ are two important issues. A related discussion is available in [14]. In fact, these approximations are strongly problem-dependent. For instance, the pressure mass matrix in Stokes problems is a natural approximation of the Schur complement and is cheaply available (see e.g. Chapter 6 in [32]), but this convenient situation does not happen in general.

Finally, we recall that the original system (1.16) involves a symmetric positive semidefinite (1,1) block G. Some block preconditioning strategies using the augmented-Lagrangian approach have been proposed. We can cite the contributions of Rees and Grief in [89] and Huang et al. in [56]. We focus here on two preconditioners issued from these references:

$$\mathcal{M}_d = \begin{pmatrix} G + B^T W B & 0 \\ 0 & W^{-1} \end{pmatrix} \text{ and } \mathcal{M}_t = \begin{pmatrix} G + B^T W B & 2B^T \\ 0 & -W^{-1} \end{pmatrix}, \tag{1.23}$$

where W is an $m \times m$ symmetric positive definite weighting matrix. As mentioned before, the matrix $G+B^T W B$ is also symmetric positive definite, as well as $\mathcal{M}_d$. Spectral studies of the preconditioned operators $\mathcal{M}_d^{-1}\mathcal{K}$ and $\mathcal{M}_t^{-1}\mathcal{K}$ have been performed in [89] and [56] respectively, and we recall the main properties in Theorem 1.3.2.

Theorem 1.3.2. *Let $\mathcal{K}$ be the saddle point matrix defined in* (1.16)*, and $\mathcal{M}_d$ and $\mathcal{M}_t$ be defined in* (1.23) *with W symmetric positive definite of order m.*

(a) The matrix $\mathcal{M}_d^{-1}\mathcal{K}$ has two distinct eigenvalues which are given by $\lambda_1 = 1$ and $\lambda_2 = -1$ with multiplicity n and $dim(\mathcal{N}(G))$, respectively. The remaining eigenvalues lie in the interval $(-1, 0)$.

(b) The matrix $\mathcal{M}_t^{-1}\mathcal{K}$ has the eigenvalues 1 with multiplicity $n + dim(\mathcal{N}(G))$. The remaining eigenvalues lie in the interval $(0, 1)$.

Both preconditioners are important in this thesis, since approximations of $\mathcal{M}_d$ and $\mathcal{M}_t$ will be used in the numerical experiments of Chapter 2 and 3, respectively. The goal is now to define W. The choice $W = \gamma I_m$ with $\gamma = \frac{\|G\|_2}{\|B\|_2^2}$ (or an approximation thereof) has been found to perform well in practice [42]. Indeed, γ can be interpreted as a coefficient to obtain an augmenting term $B^T W B$ of norm of similar magnitude in

comparison with G. This scaling strategy can be related to the choice of the coefficient α in (1.22) for the "double Lagrange" method used in Code_Aster. Actually, we will choose $\gamma = \alpha$ in our numerical tests, since this coefficient is immediately available. We also note that B is a very sparse matrix in our setting. As explained in Section 1.1.1, B is related to the dualization of single or multifreedom constraints. These relations are local in the sense that they involve either one node or adjacent nodes of the mesh. If B contains only essential boundary conditions, B admits one nonzero coefficient per row and $B^T B$ is a diagonal matrix. Thus, the matrix $\gamma B^T B$ does not degrade significantly the sparsity pattern of $G + \gamma B^T B$.

1.4 Conclusions

In this chapter, we have carried out a presentation of the mathematical formulation coming from solid mechanics problems, and have provided a short introduction of Krylov subspace methods. Particularly, we have focused on the solution of saddle point systems and some attractive preconditioners have been detailed. Since the framework of this thesis is related to the solution of sequences of saddle point systems, the following chapters focus more generally on improvement of preconditioning techniques for symmetric indefinite and nonsymmetric systems, respectively.

Chapter 2

Limited memory preconditioners for symmetric indefinite systems

This chapter states our main contribution in proposing and studying a class of limited memory preconditioners (named $\text{LMP}_{\pm}$), adapted to solve a sequence of linear algebraic symmetric indefinite systems with a Krylov subspace method. We first analyse this $\text{LMP}_{\pm}$ class theoretically and then we illustrate its efficiency on systems of saddle point structure arising in solid mechanics, as described in Section 1.1.

When the coefficient matrices in a sequence are symmetric positive definite, Morales and Nocedal [71] have proposed a preconditioner which has the form of a limited memory quasi-Newton matrix. This automatic preconditioner generated using information from the conjugate gradient method [65, 66] does not require explicit knowledge of the coefficient matrix and is therefore suitable for problems where only products of the matrix times a vector can be computed, as in data assimilation. The contribution [71] extends earlier attempts in this direction; see, e.g., [75, 80]. More recently, Gratton, Sartenaer and Tshimanga [46] have defined a class of limited memory preconditioners (LMP) based on limited memory quasi-Newton formulas that ensures good spectral properties of the preconditioned matrix. These preconditioners require a small number k of linearly independent vectors and may also be used to improve an existing (possibly application dependent) first-level preconditioner. This family can be seen as a block variant of the BFGS updating formula for quadratic problems [78, 93]. A spectral analysis of the preconditioned matrix has shown that the LMP class is able to cluster at least k eigenvalues at 1 and that the eigenvalues of the preconditioned matrix enjoy interlacing properties with respect to the eigenvalues of the original matrix. The efficiency of the precondi-

tioner has been shown on a real-life application in data assimilation [46, 109].

Our main objective here is to propose an extension of the limited memory preconditioners to be used when the coefficient matrices are symmetric indefinite. To the best of our knowledge, we are unaware of any proposition in this direction. The chapter is organized as follows. In Section 2.1, we first present the class of LMPs developed in the symmetric positive definite case studied in [46] and [109]. We extend their definition to the symmetric negative definite case. In Section 2.2, the main theoretical section of the chapter, we extend the class of LMPs to the symmetric indefinite case (called LMP$_\pm$) and expose our three main contributions. First, we derive a formula to characterize the spectrum of the preconditioned operator. Secondly, we show that the eigenvalues of the preconditioned matrix enjoy interlacing properties with respect to the eigenvalues of the original matrix provided that the k vectors have been prior projected onto the invariant subspaces associated with the eigenvalues of the original matrix in the open right and left half-plane, respectively. Third, we focus on theoretical properties of the Ritz-LMP$_\pm$ variant, where Ritz information is used to determine the k vectors. After discussing some implementation issues, we explore in Section 2.4 the numerical performance of the limited memory preconditioner used to improve an existing first-level preconditioner on possibly large-scale applications in solid mechanics, as introduced before in this manuscript.

2.1 Limited memory preconditioners for symmetric definite matrices

In this section, we briefly review the main properties of the limited memory preconditioners for linear systems involving symmetric (positive or negative) definite matrices [46].

2.1.1 Limited memory preconditioners for symmetric positive definite matrices

Many problems in computational science and engineering require the solution of a sequence of linear systems of type $Ax_i = b_i, i = 1, \ldots, I$ with $A \in \mathbb{R}^{N\times N}$ being symmetric positive definite, $x_i \in \mathbb{R}^N$ and $b_i \in \mathbb{R}^N$. For large-scale problems the conjugate gradient method [54] is generally the method of choice for solving such a sequence, where A could represent either the original or an already preconditioned operator. The convergence behaviour of the conjugate gradient method can be potentially improved with the notion

of limited memory preconditioner defined next [46, Definition 2.1].

Definition 2.1.1. *Let A be a symmetric positive definite matrix of order N and assume that $S \in \mathbb{R}^{N\times k}$ is of full column rank, with $k \leq N$. The symmetric matrix H of order N defined as*

$$H = (I_N - S(S^TAS)^{-1}S^TA)(I_N - AS(S^TAS)^{-1}S^T) + S(S^TAS)^{-1}S^T \tag{2.1}$$

is called the limited memory preconditioner (LMP).

This family can be seen as a block variant of the BFGS updating formula for quadratic problems [78, 93]. Actually, let define the strictly convex function

$$q(x) = x^TAx - x^Tb,$$

where $A \in \mathbb{R}^{N\times N}$ is symmetric positive definite, x and $b \in \mathbb{R}^N$. During the BFGS process at iterate k, an approximation of the inverse of the Hessian of q (i.e. A^{-1}) is obtained as

$$H_k = (I_N - \frac{s_ky_k^T}{y_k^Ts_k})H_{k-1}(I_N - \frac{y_ks_k^T}{y_k^Ts_k}) + \frac{s_ks_k^T}{y_k^Ts_k}. \tag{2.2}$$

In this expression, $s_k = x_k - x_{k-1}$ is obtained from a line search along the descent direction and $y_k = \nabla q(x_k) - \nabla q(x_{k-1}) = As_k$. The block generalization yielding Definition 2.1 is detailed in [46]. This relation is important here, since the different preconditioning update techniques developed in this manuscript are based on optimization methods (see Section 3.1 for the nonsymmetric case).

H defined by (2.1) is a symmetric positive definite preconditioner [46, Lemma 3.3] satisfying $HAS = S$, i.e. the limited memory preconditioner is able to cluster at least k eigenvalues of HA at 1. In addition, the eigenvalues of the preconditioned matrix HA enjoy interlacing properties with respect to the eigenvalues of the original matrix A. This central result on the clustering of the spectrum of the preconditioned matrix HA is stated in Theorem 2.1.1 given next [46, Theorem 3.4].

Theorem 2.1.1. *Let the positive real numbers $\sigma_1, \cdots, \sigma_N$ denote the eigenvalues of A sorted in non decreasing order. Then the set of eigenvalues $\mu_1, \cdots, \mu_N$ of HA can be*

split in two subsets

$$\begin{aligned} \sigma_j \leq \mu_j \leq \sigma_{j+k} \qquad & \textit{for } j \in \{1, \cdots, N-k\}, \\ \mu_j = 1 \qquad & \textit{for } j \in \{N-k+1, \cdots, N\}. \end{aligned}$$

In addition, the condition number of HA can be bounded as follows

$$\frac{\max_{j=1,\cdots,N} \mu_j}{\min_{j=1,\cdots,N} \mu_j} \leq \frac{\max\{1, \sigma_N\}}{\min\{1, \sigma_1\}}. \tag{2.3}$$

Figures 2.1 and 2.2 give two illustrations of this theorem, for random symmetric positive definite matrices $A \in \mathbb{R}^{30\times 30}$ and a random matrix $S \in \mathbb{R}^{30\times 5}$. Figure 2.1 shows a case where 1 lies in $[\sigma_1, \sigma_N]$, while Figure 2.2 is related to the situation where the eigenvalues of A are larger than 1. The corresponding spectral property in the symmetric indefinite case is further discussed in Section 2.2.4.

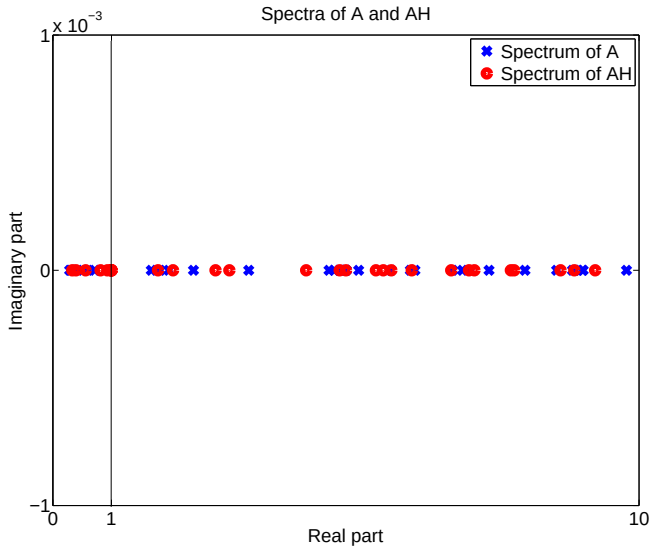

Figure 2.1: Eigendistribution of A and AH: case of $1 \in [\sigma_1, \sigma_N]$.

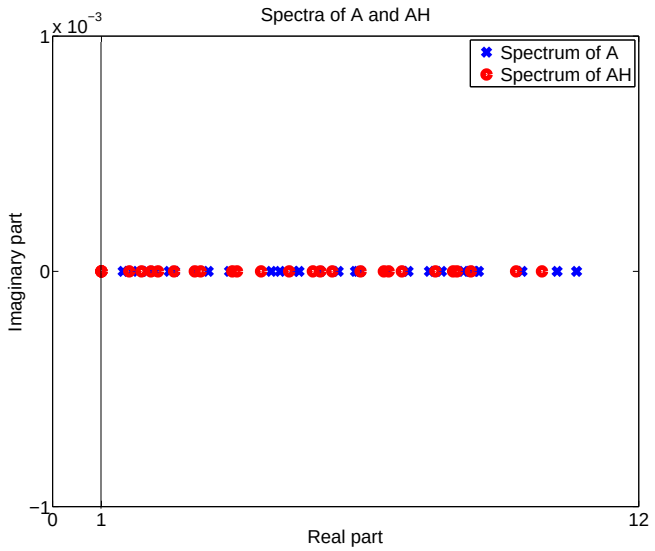

Figure 2.2: Eigendistribution of A and AH: case of eigenvalues of A larger than 1.

2.1.2 Limited memory preconditioners for symmetric negative definite matrices

As required later in Section 2.2, we consider the extension of limited memory preconditioners to the case of symmetric negative definite matrices. A straightforward adaptation of Theorem 2.1.1 is given next as a corollary.

Corollary 2.1.2. *Let A be a symmetric negative definite matrix of order N and assume that $S \in \mathbb{R}^{N\times k}$ is of full column rank, with $k \leq N$. Let H denote a symmetric matrix of order N given by (2.1) in Definition 2.1.1. Let the negative real numbers $\sigma_1, \cdots, \sigma_N$ denote the eigenvalues of A, sorted in non decreasing order. Then the set of eigenvalues $\mu_1, \cdots, \mu_N$ of HA can be split in two subsets*

$$\begin{aligned} \sigma_j \leq \mu_j \leq \sigma_{j+k} \qquad & \text{for } j \in \{1, \cdots, N-k\}, \\ \mu_j = 1 \qquad & \text{for } j \in \{N-k+1, \cdots, N\}. \end{aligned}$$

In addition, the condition number of HA can be bounded as follows

$$\frac{\max_{j=1,\cdots,N} |\mu_j|}{\min_{j=1,\cdots,N} |\mu_j|} \leq \frac{\max\{1, |\sigma_1|\}}{\min\{1, |\sigma_N|\}}. \tag{2.4}$$

2.2 Limited memory preconditioners for symmetric indefinite matrices

In this section, we expose the main theoretical properties of the proposed limited memory preconditioners applied to the solution of symmetric indefinite linear systems.

2.2.1 Definition

We address the solution of sequence of linear systems of type

$$Ax_i = b_i \quad i = 1, \ldots, I \tag{2.5}$$

with $A \in \mathbb{R}^{N\times N}$ being symmetric indefinite, $x_i \in \mathbb{R}^N$ and $b_i \in \mathbb{R}^N$. We emphasize that the matrix is considered to be fixed in the forthcoming analysis. Nevertheless, we will also use the proposed preconditioning technique, defined next, for sequences with slowly varying matrices (see Sections 2.4.4 and 2.4.5).

Definition 2.2.1. *Let A be a symmetric indefinite matrix of order N. Assume that $S \in \mathbb{R}^{N\times k}$, with $k \leq N$, is such that S^TAS is nonsingular. The symmetric matrix H defined as*

$$H = (I_N - S(S^TAS)^{-1}S^TA)(I_N - AS(S^TAS)^{-1}S^T) + S(S^TAS)^{-1}S^T \tag{2.6}$$

is called the limited memory preconditioner in the indefinite case (LMP$_\pm$).

This definition is quite close to Definition 2.1.1. The difference is related to the assumption about S. When A is symmetric indefinite, the linear independence of the columns of S is not sufficient to insure the nonsingularity of S^TAS. Even the simple case $k = 1$ can be problematic: if $A = \mathrm{diag}(1, -1)$ and $S = [1, 1]^T$, we obtain $S^TAS = 0$.

Before studying the spectrum of the preconditioned operator AH, we give a first useful property of the LMP$_\pm$ as a straightforward adaptation of Theorem 3.1 in [46]. Proposition 2.2.1 shows the invariance of H defined by (2.6) under a change of basis of $\mathcal{S} = \mathcal{R}(S)$.

Proposition 2.2.1. *Let $S \in \mathbb{R}^{N\times k}$ such that S^TAS is nonsingular. If $Z = SX$, where X is a $k \times k$ invertible matrix, replacing S by Z in* (2.6) *does not change H.*

2.2.2 Spectrum of AH

Now, we aim at characterizing the spectrum of the preconditioned operator AH in the indefinite case. This first contribution is stated in Theorem 2.2.2.

Theorem 2.2.2. *Let A be a symmetric indefinite matrix of order N and H be given by (2.6) in Definition 2.2.1. Assume that the columns of $Z \in \mathbb{R}^{N\times k}$ form an orthonormal basis for $\mathcal{S}$ and that the columns of $Z_\perp \in \mathbb{R}^{N\times(N-k)}$ form an orthonormal basis for $\mathcal{S}^\perp$. The spectrum of the preconditioned operator AH is then given by*

$$\Lambda(AH) = \{1\} \cup \Lambda((Z_\perp^T A^{-1} Z_\perp)^{-1}).$$

Proof. A direct calculation leads to

$$AH = P_{\mathcal{S}^\perp,A\mathcal{S}} A P_{\mathcal{S}^\perp,A\mathcal{S}} + I_N - P_{\mathcal{S}^\perp,A\mathcal{S}},$$

with $P_{\mathcal{S}^\perp,A\mathcal{S}} = I_N - AS(S^TAS)^{-1}S^T$ the oblique projection onto $\mathcal{S}^\perp$ along $A\mathcal{S}$. To determine the spectrum of AH we consider the matrix $[Z, Z_\perp]^T\ AH\ [Z, Z_\perp]$ which is congruent to AH

$$[Z, Z_\perp]^T\ AH\ [Z, Z_\perp] = \begin{pmatrix} Z^TAHZ & Z^TAHZ_\perp \\ Z_\perp^TAHZ & Z_\perp^TAHZ_\perp \end{pmatrix}.$$

Since $\mathcal{R}(P_{\mathcal{S}^\perp,A\mathcal{S}}) = \mathcal{S}^\perp$, $\mathcal{N}(I_N - P_{\mathcal{S}^\perp,A\mathcal{S}}) = \mathcal{S}^\perp$ and the fact that Z has orthonormal columns, we obtain

$$[Z, Z_\perp]^T\ AH\ [Z, Z_\perp] = \begin{pmatrix} I_k & 0_{k,N-k} \\ Z_\perp^TAHZ & Z_\perp^T P_{\mathcal{S}^\perp,A\mathcal{S}} A Z_\perp \end{pmatrix}. \tag{2.7}$$

Hence we deduce that 1 is an eigenvalue of AH at least of multiplicity k. Moreover, the spectrum of $P_{\mathcal{S}^\perp,A\mathcal{S}}A$ can be characterized via the inverse of A as recently shown in [38, Corollary 3.25]

$$\Lambda(P_{\mathcal{S}^\perp,A\mathcal{S}}A) = \{0\} \cup \Lambda((Z_\perp^T A^{-1} Z_\perp)^{-1}).$$

Since $\mathcal{N}(P_{\mathcal{S}^\perp,A\mathcal{S}}A) = \mathcal{S}$, we obtain

$$\Lambda(Z_\perp^T P_{\mathcal{S}^\perp,A\mathcal{S}} A Z_\perp) = \Lambda((Z_\perp^T A^{-1} Z_\perp)^{-1}).$$

Finally, $[Z, Z_\perp]$ being orthonormal, we have $\Lambda(AH) = \Lambda([Z, Z_\perp]^T\ AH\ [Z, Z_\perp])$ and the

proof is complete. □

Theorem 2.2.2 is valid for any set of k linearly independent vectors such that S^TAS is nonsingular. We further note that the characterization of the spectrum of AH given in Theorem 2.2.2 holds for any invertible operator A, see [38, Theorem 3.24]. As shown in Theorem 2.2.2, the eigenvalues of AH are located on the real axis. The question of the sign of the eigenvalues of AH is addressed more precisely next.

2.2.3 Sign of the eigenvalues of AH and inertia of H

In light of (2.7), the spectrum of AH is made of at least k eigenvalues equal to 1 and of eigenvalues of $Z_\perp^T P_{\mathcal{S}^\perp,A\mathcal{S}} A Z_\perp$. Hence, to characterize the sign of the eigenvalues of AH, we need to determine the inertia of $Z_\perp^T P_{\mathcal{S}^\perp,A\mathcal{S}} A Z_\perp$ as stated next in Theorem 2.2.3. We recall that the inertia of a symmetric matrix B is a triplet of nonnegative integers (denoted as $\mathrm{In}(B) = (m, z, p)$), where m, z and p are the number of negative, zero, and positive elements of $\Lambda(B)$ [43].

Theorem 2.2.3. *Let A be a symmetric indefinite matrix of order N and H be given by (2.6) in Definition 2.2.1. The inertia of $Z_\perp^T P_{\mathcal{S}^\perp,A\mathcal{S}} A Z_\perp$ is then given by*

$$In(Z_\perp^T P_{\mathcal{S}^\perp,A\mathcal{S}} A Z_\perp) = In(A) - In(S^TAS).$$

Proof. We consider the symmetric matrix B defined as

$$B = [Z, Z_\perp]^T \; A \; [Z, Z_\perp],$$

or equivalently

$$B = \begin{pmatrix} Z^TAZ & Z^TAZ_\perp \\ Z_\perp^TAZ & Z_\perp^TAZ_\perp \end{pmatrix}.$$

We remark that B and A have the same inertia due to Sylvester's law of inertia [43]. Since Z^TAZ is nonsingular due to Definition 2.2.1, we can apply the Haynsworth inertia additivity formula [52] to obtain

$$In(B) = In(Z^TAZ) + In(Z_\perp^TAZ_\perp - Z_\perp^TAZ(Z^TAZ)^{-1}Z^TAZ_\perp),$$

that also reads

$$In(B) = In(Z^TAZ) + In(Z_\perp^T P_{\mathcal{Z}^\perp,A\mathcal{Z}} A Z_\perp).$$

Since Z^TAZ and S^TAS have the same inertia and $P_{\mathcal{Z}^\perp,A\mathcal{Z}}A = P_{\mathcal{S}^\perp,A\mathcal{S}}A$, we obtain the final result. □

Theorem 2.2.3 is helpful if the symmetric and indefinite operator A admits l negative eigenvalues where $l \leq k << N$. In such a case, if $S \in \mathbb{R}^{N\times k}$ is known such that S^TAS admits l negative eigenvalues, Theorem 2.2.3 states that AH admits only real positive eigenvalues. We conclude this section by characterizing the inertia of H.

Theorem 2.2.4. *Let A be a symmetric indefinite matrix of order N and H be given by (2.6) in Definition 2.2.1. Assume that the columns of $W \in \mathbb{R}^{N\times k}$ form an orthonormal basis for $A\mathcal{S}$ and that the columns of $W_\perp \in \mathbb{R}^{N\times(N-k)}$ form an orthonormal basis for $(A\mathcal{S})^\perp$. The inertia of H is then given by*

$$In(H) = In(S^TAS) + In(W_\perp^T P_{\mathcal{S}^\perp,A\mathcal{S}}^T P_{\mathcal{S}^\perp,A\mathcal{S}} W_\perp).$$

Proof. Similarly as in Theorem 2.2.3, we consider the symmetric matrix C defined as

$$C = [W, W_\perp]^T \; H \; [W, W_\perp].$$

C and H have the same inertia due to Sylvester's law of inertia. Furthermore, since the columns of W form an orthonormal basis for $A\mathcal{S}$, it exists a nonsingular matrix $X \in \mathbb{R}^{k\times k}$ such that $W = ASX$. We obtain after calculation

$$C = \begin{pmatrix} X^TS^TASX & X^TS^TW_\perp \\ W_\perp^TSX & W_\perp^THW_\perp \end{pmatrix}.$$

By applying the Haynsworth inertia additivity formula on C and the Sylvester's law of inertia on X^TS^TASX, we have

$$\begin{aligned} In(H) &= In(S^TAS) + In(W_\perp^THW_\perp - W_\perp^TSX(X^TS^TASX)^{-1}X^TS^TW_\perp) \\ &= In(S^TAS) + In(W_\perp^T(H - S(S^TAS)^{-1}S^T)W_\perp) \end{aligned}$$

Since

$$H = P_{\mathcal{S}^\perp,A\mathcal{S}}^T P_{\mathcal{S}^\perp,A\mathcal{S}} + S(S^TAS)^{-1}S^T,$$

the proof is complete. □

An important consequence of Theorem 2.2.4 is that the number of negative eigenvalues of H is equal to the number of negative eigenvalues of S^TAS, since $W_\perp^T P_{\mathcal{S}^\perp,A\mathcal{S}}^T P_{\mathcal{S}^\perp,A\mathcal{S}} W_\perp$

is symmetric positive definite.

2.2.4 Nonexpansion of the spectrum of AH

In this section, we investigate the question related to the nonexpansion of the spectrum of AH. In general, this property does not hold any longer in the indefinite case as illustrated by this simple example

$$A = \begin{pmatrix} 2 & 0 \\ 0 & -1 \end{pmatrix}, \quad S = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \quad H = \begin{pmatrix} 3 & 5 \\ 5 & 9 \end{pmatrix}, \quad \Lambda(AH) = \{-4\} \cup \{1\}.$$

The second contribution of this chapter is to show that a nonexpansion property of the spectrum of AH holds provided that the k linearly independent vectors defining S have been prior projected onto the *invariant* subspaces associated with the eigenvalues of A in the open right and left half-plane, respectively. These projection operators involving the matrix sign function of A are defined next [55].

Definition 2.2.2. *Let $A \in \mathbb{R}^{N\times N}$ be a symmetric indefinite matrix of order N and let $\mathrm{sign}(A) \in \mathbb{R}^{N\times N}$ denote the matrix sign function*[1] *of A defined as $\mathrm{sign}(A) = (A^2)^{-\frac{1}{2}}A$. Let $\mathcal{I}_+(A)$ and $\mathcal{I}_-(A)$ denote the invariant subspaces associated with the eigenvalues in the right and left half-plane, respectively. We define $P_+(A) = (I_N + \mathrm{sign}(A))/2$ as the projection operator onto $\mathcal{I}_+(A)$ and $P_-(A) = (I_N - \mathrm{sign}(A))/2$ as the projection operator onto $\mathcal{I}_-(A)$, respectively.*

We denote by $Q_+ \in \mathbb{R}^{N\times N_+}$ ($Q_- \in \mathbb{R}^{N\times N_-}$) an orthonormal basis of $\mathcal{I}_+(A)$ ($\mathcal{I}_-(A)$, respectively) and by $Q \in \mathbb{R}^{N\times N}$ the orthonormal matrix defined as $Q = [Q_+, Q_-]$ with $N = N_+ + N_-$. Given $\tilde{S} \in \mathbb{R}^{N\times k}$, $S = [S_+, S_-]$ ($S_+ \in \mathbb{R}^{N\times k_+}$, $S_- \in \mathbb{R}^{N\times k_-}$ with $k = k_+ + k_-, k \leq N$) consists of k projected vectors obtained as

$$S_+ = Q_+Q_+^T[\tilde{s}_{i_1}, ..., \tilde{s}_{i_{k_+}}], \tag{2.8}$$

$$S_- = Q_-Q_-^T[\tilde{s}_{j_1}, ..., \tilde{s}_{j_{k_-}}], \tag{2.9}$$

where $[\tilde{s}_{i_1}, ..., \tilde{s}_{i_{k_+}}]$ ($[\tilde{s}_{j_1}, ..., \tilde{s}_{j_{k_-}}]$) corresponds to k_+ (k_-, respectively) distinct columns

1. A (being symmetric indefinite) has no eigenvalues on the imaginary axis, so that the matrix sign function of A is defined.

of $\tilde{S}$. Equivalently, we can write:

$$S_+ = Q_+ \tilde{S}_+, \quad \tilde{S}_+ \in \mathbb{R}^{N_+ \times k_+}, \tag{2.10}$$

$$S_- = Q_- \tilde{S}_-, \quad \tilde{S}_- \in \mathbb{R}^{N_- \times k_-}. \tag{2.11}$$

The main goal of the next developments is to show that a property of nonexpansion of the spectrum of HA can be obtained by solving two tractable subproblems related to either $\mathcal{I}_+(A)$ or $\mathcal{I}_-(A)$. We first prove that $\mathcal{I}_+(A)$ and $\mathcal{I}_-(A)$ are H-invariant, by showing Lemma 2.2.5 and Lemma 2.2.6 successively.

Lemma 2.2.5. *Define $T \in \mathbb{R}^{N\times N}$ as $T = S(S^TAS)^{-1}S^T$, $T_+ \in \mathbb{R}^{N\times N}$ as $T_+ = S_+(S_+^TAS_+)^{-1}S_+^T$ and $T_- \in \mathbb{R}^{N\times N}$ as $T_- = S_-(S_-^TAS_-)^{-1}S_-^T$, respectively. T can be decomposed as*

$$T = T_+ + T_-.$$

Proof. Since $\mathcal{I}_+(A)$ and $\mathcal{I}_-(A)$ are A-invariant and orthogonal subspaces, the relation $S_-^TAS_+ = 0_{k_-,k_+}$ holds. Hence S^TAS can be written as

$$S^TAS = [S_+, S_-]^TA[S_+, S_-] = \begin{pmatrix} S_+^TAS_+ & 0_{k_+,k_-} \\ 0_{k_-,k_+} & S_-^TAS_- \end{pmatrix}.$$

Since S^TAS is assumed to be nonsingular, $S_+^TAS_+$ and $S_-^TAS_-$ are also nonsingular and we deduce

$$\begin{aligned} S(S^TAS)^{-1}S^T &= [S_+, S_-] \begin{pmatrix} (S_+^TAS_+)^{-1} & 0_{k_+,k_-} \\ 0_{k_-,k_+} & (S_-^TAS_-)^{-1} \end{pmatrix} [S_+, S_-]^T, \\ &= S_+(S_+^TAS_+)^{-1}S_+^T + S_-(S_-^TAS_-)^{-1}S_-^T, \end{aligned}$$

which completes the proof. □

Lemma 2.2.6. *$\mathcal{I}_+(A)$ is H-invariant*

$$\forall v \in \mathcal{I}_+(A) \quad Hv \in \mathcal{I}_+(A). \tag{2.12}$$

Proof. Due to Lemma 2.2.5 and orthogonality of $\mathcal{I}_+(A)$ and $\mathcal{I}_-(A)$, we obtain

$$\forall v \in \mathcal{I}_+(A) \quad Tv = T_+v,$$

meaning that $Tv \in \mathcal{I}_+(A)$. We also deduce that $ATv \in \mathcal{I}_+(A)$ since $\mathcal{I}_+(A)$ is A-invariant.

Furthermore Hv can be simply written as

$$Hv = (I_N - TA)(I_N - AT)v + Tv.$$

Hence we deduce relation (2.12) i.e. $\mathcal{I}_+(A)$ is H-invariant. We also note that a similar proof leads to the H-invariance of $\mathcal{I}_-(A)$. □

Lemma 2.2.7 states a similarity property that is central in the analysis of $\Lambda(HA)$.

Lemma 2.2.7. *Let $A_+ = Q_+^T A Q_+ \in \mathbb{R}^{N_+ \times N_+}$ ($A_- = Q_-^T A Q_- \in \mathbb{R}^{N_- \times N_-}$) denote the orthogonally projected restriction of A with respect to the basis Q_+ (Q_-, respectively). Let $H_+ = Q_+^T H Q_+ \in \mathbb{R}^{N_+ \times N_+}$ ($H_- = Q_-^T H Q_- \in \mathbb{R}^{N_- \times N_-}$) denote the orthogonally projected restriction of H with respect to the basis Q_+ (Q_-, respectively). Then $Q^T HAQ$ admits the following decomposition*

$$Q^T HAQ = \begin{pmatrix} H_+A_+ & 0_{N_+,N_-} \\ 0_{N_-,N_+} & H_-A_- \end{pmatrix}.$$

As a consequence, $\Lambda(HA) = \Lambda(H_+A_+) \cup \Lambda(H_-A_-)$.

Proof. Since $\mathcal{I}_+(A)$ and $\mathcal{I}_-(A)$ are A-invariant and orthogonal subspaces, the relation $Q_-^T A Q_+ = 0_{N_-,N_+}$ holds. $Q^T AQ$ can then be written as

$$Q^T AQ = \begin{pmatrix} A_+ & 0_{N_+,N_-} \\ 0_{N_-,N_+} & A_- \end{pmatrix}.$$

Furthermore due to the H-invariance of $\mathcal{I}_+(A)$ (Lemma 2.2.6) and the orthogonality of Q, we deduce that $Q_-^T H Q_+ = 0_{N_-,N_+}$. Thus we obtain

$$Q^T HQ = \begin{pmatrix} H_+ & 0_{N_+,N_-} \\ 0_{N_-,N_+} & H_- \end{pmatrix},$$

which leads to:

$$Q^T HAQ = \begin{pmatrix} H_+A_+ & 0_{N_+,N_-} \\ 0_{N_-,N_+} & H_-A_- \end{pmatrix}.$$

This similarity relation immediately implies that $\Lambda(HA) = \Lambda(H_+A_+) \cup \Lambda(H_-A_-)$. □

Consequently, we must now focus on the analysis of H_+A_+ and H_-A_-, respectively. In Lemma 2.2.8, we show that H_+ and H_- are both of limited memory preconditioner

type.

Lemma 2.2.8. *Define* $\tilde{T}_+ \in \mathbb{R}^{N_+ \times N_+}$ *as* $\tilde{T}_+ = \tilde{S}_+(\tilde{S}_+^T A_+ \tilde{S}_+)^{-1}\tilde{S}_+^T$ *and* $\tilde{T}_- \in \mathbb{R}^{N_- \times N_-}$ *as* $\tilde{T}_- = \tilde{S}_-(\tilde{S}_-^T A_- \tilde{S}_-)^{-1}\tilde{S}_-^T$*, respectively.* H_+ *and* H_- *can be written as*

$$H_+ = (I_{N_+} - \tilde{T}_+ A_+)(I_{N_+} - A_+\tilde{T}_+) + \tilde{T}_+, \tag{2.13}$$
$$H_- = (I_{N_-} - \tilde{T}_- A_-)(I_{N_-} - A_-\tilde{T}_-) + \tilde{T}_-. \tag{2.14}$$

As a consequence, H_+ *and* H_- *are both limited memory preconditioners.*

Proof. Using successively Lemma 2.2.5, the definition of $\tilde{T}_+$ and A_+, and the orthogonality of Q, we obtain

$$\begin{aligned}
H_+ &= (Q_+^T - Q_+^T T_+ A)(Q_+ - AT_+Q_+) + Q_+^T T_+ Q_+, \\
H_+ &= (Q_+^T - \tilde{T}_+ Q_+^T A)(Q_+ - AQ_+\tilde{T}_+) + \tilde{T}_+, \\
H_+ &= (Q_+^T - \tilde{T}_+ Q_+^T A)QQ^T(Q_+ - AQ_+\tilde{T}_+) + \tilde{T}_+, \\
H_+ &= (I_{N_+} - \tilde{T}_+ A_+)(I_{N_+} - A_+\tilde{T}_+) + \tilde{T}_+.
\end{aligned}$$

Relation (2.14) can be obtained by a similar proof. Since A_+ is symmetric positive definite, relation (2.13) reveals that H_+ is a limited memory preconditioner related to the symmetric positive definite case (see Definition 2.1.1). Similarly, A_- being symmetric positive negative, H_- defines a limited memory preconditioner related to the symmetric negative definite case (see Corollary 2.1.2). □

Based on the previous developments, we finally state the main result related to the nonexpansion of the spectrum of HA.

Theorem 2.2.9. *Let* A *be a symmetric indefinite matrix of order* N*,* H *be given by (2.6) in Definition 2.2.1 based on* $S = [S_+, S_-]$ *consisting of* k_+ *(*k_-*) vectors projected onto the positive (negative, respectively) invariant subspace of* A*,* $\mathcal{I}_+(A)$ *(*$\mathcal{I}_-(A)$*, respectively). Then the following properties hold*

(a) Let the positive real numbers $\sigma_1^+, \cdots, \sigma_{N_+}^+$ *denote the eigenvalues of* A_+ *sorted in nondecreasing order. Then the set of eigenvalues* $\mu_1^+, \cdots, \mu_{N_+}^+$ *of* H_+A_+ *can be split in two subsets*

$$\begin{aligned}
\sigma_j^+ \leq \mu_j^+ \leq \sigma_{j+k_+}^+ \text{ for } j \in \{1, \cdots, N_+ - k_+\}, \\
\mu_j^+ = 1 \text{ for } j \in \{N_+ - k_+ + 1, \cdots, N_+\}.
\end{aligned} \tag{2.15}$$

(b) Let the negative real numbers $\sigma_1^-, \cdots, \sigma_{N_-}^-$ *denote the eigenvalues of* A_- *sorted in nondecreasing order. Then the set of eigenvalues* $\mu_1^-, \cdots, \mu_{N_-}^-$ *of* $H_- A_-$ *can be split in two subsets*

$$\begin{aligned} \sigma_j^- \leq \mu_j^- \leq \sigma_{j+k_-}^- \text{ for } j \in \{1, \cdots, N_- - k_-\}, \\ \mu_j^- = 1 \text{ for } j \in \{N_- - k_- + 1, \cdots, N_-\}. \end{aligned} \tag{2.16}$$

Proof. Since H_+ and H_- are limited memory preconditioners (see Lemma 2.2.8), Properties (a) and (b) are direct consequences of Theorem 2.1.1 and Corollary 2.1.2, respectively. □

We illustrate Theorem 2.2.9 on two examples with random symmetric indefinite matrices $A \in \mathbb{R}^{30\times 30}$ and a random matrix $S \in \mathbb{R}^{30\times 5}$, where the columns of S are prior projected onto $\mathcal{I}_+(A)$ or $\mathcal{I}_-(A)$. In Figure 2.3, the positive eigenvalues of A satisfy $\sigma_1^+ \leq 1 \leq \sigma_{N_+}^+$, contrary to the case illustrated in Figure 2.4. Both reveal for $\Lambda(AH)$ a cluster at 1 and a contraction of the remaining eigenvalues relatively to $\Lambda(A)$ on each side of the plane.

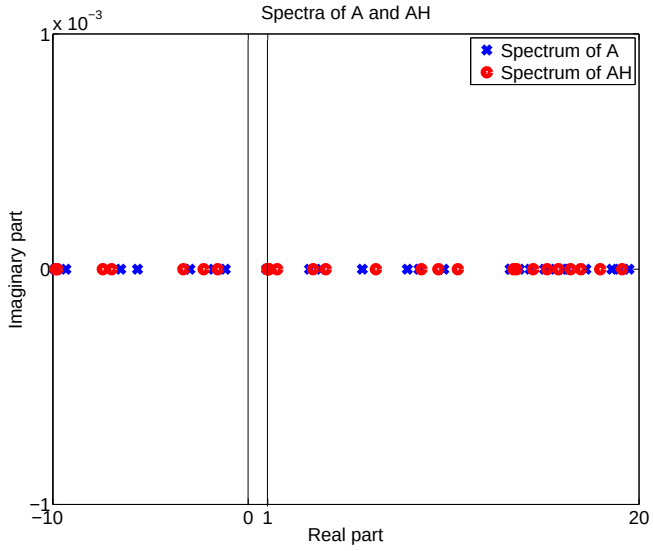

Figure 2.3: Eigendistribution of A and AH: case of $1 \in [\sigma_1^+, \sigma_{N_+}^+]$.

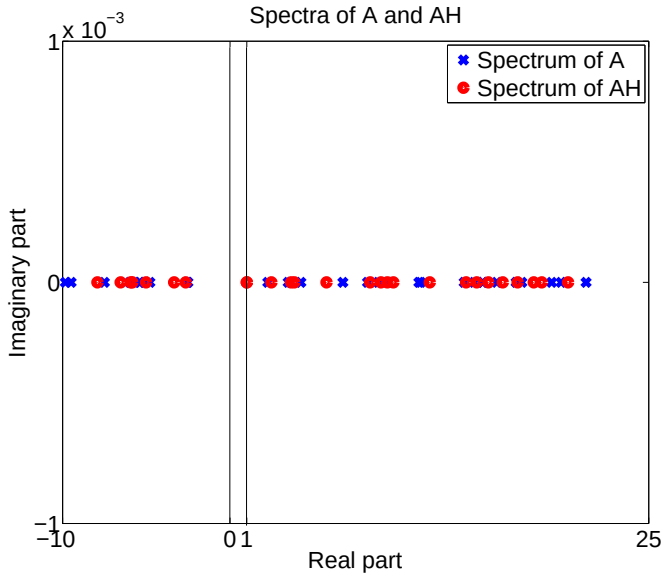

Figure 2.4: Eigendistribution of A and AH: case of positive eigenvalues of A larger than 1.

2.2.5 Ritz limited memory preconditioner

As shown in Theorem 2.2.9, the use of projected vectors in the limited memory preconditioner insures a nonexpansion property of the spectrum of the preconditioned operator, which is an attractive feature. Nevertheless, using the exact sign function of A can be computationally too expensive for large-scale problems. In the literature, some methods exist to approximate $\text{sign}(A)f$ where $f \in \mathbb{R}^N$. A standard approach consists in computing a Lanczos method applied to the matrix A with initial vector f (see e.g. [36] and the references therein). According to Section 1.2.1, we obtain a Lanczos relation of the form

$$A\hat{V}_l = \hat{V}_l\hat{T}_l + \hat{v}_{l+1}(\hat{t}_{l+1,l}e_l^T).$$

The Lanczos approximation of $\text{sign}(A)f$ from this Lanczos relation is then defined by

$$\bar{f}_l = \hat{V}_l\text{sign}(\hat{T}_l)\hat{V}_l^T f. \tag{2.17}$$

The idea behind this approximation is that $\hat{T}_l$ represents the compression of the matrix A onto $\mathcal{K}_l(A, f)$ with respect to the basis $\hat{V}_l$. Moreover, since f is the initial vector of

the Lanczos process, we have $f = \beta \hat{V}_l e_1$ with $\beta = f/\|f\|_2$ and

$$\bar{f}_l = \beta \hat{V}_l \text{sign}(\hat{T}_l) e_1.$$

Different variants of this method have been studied. For instance, the authors in [18] use a nested Krylov subspace method to improve the approximation of $\text{sign}(\hat{T}_l)$. In [37], the method is adapted when the sign function $t \to sign(t)$ is approximated by a rational function, the Zolotarev approximation.

In order to avoid an overcost related to the treatment of the column vectors of S in (2.6) as introduced above, we have chosen a different approach in this study. Indeed, we know from Section 1.2.3 that it is possible to cheaply recover approximate spectral information during the solution of the first linear system of (2.5), thanks to Ritz or harmonic Ritz vectors. Being approximations of eigenvectors of A, they are "close" to belong to the invariant subspaces $\mathcal{I}_+(A)$ or $\mathcal{I}_-(A)$ (depending on the sign of the associated Ritz or harmonic Ritz values). Between the two possibilities, we will see in Section 2.3 that the use of Ritz vectors is more interesting in terms of computational cost and memory requirements. Moreover, in terms of iteration count of GMRES, the gains are very similar in our numerical experiments, as illustrated in Section 2.4.2. We analyse next the variant of the limited memory preconditioner called Ritz limited memory preconditioner (Ritz-LMP$_\pm$).

2.2.5.1 Characterization of the Ritz vectors

We consider here the solution of the first linear system $Ax_1 = b_1$ of the sequence (2.5) with a Krylov subspace method. The application of the Lanczos method described in Section 1.2.1 leads to the Lanczos relation

$$AV_l = V_l T_l + v_{l+1}(t_{l+1,l} e_l^T), \tag{2.18}$$

where $V_l = [v_1, \cdots, v_l]$ has orthonormal columns and $T_l \in \mathbb{R}^{l \times l}$ is a symmetric tridiagonal matrix. As mentioned in Section 1.2.3, determining the Ritz pairs of A with respect to $\mathcal{R}(V_l)$ requires the solution of the standard eigenvalue problem

$$T_l Y = Y \Theta \tag{2.19}$$

with $Y^TY = I_l$ since T_l is symmetric, $Y = [y_1, \cdots, y_l]$ and $\Theta_k = diag(\theta_1, \cdots, \theta_l)$. Given $1 \leq k \leq l$, we select k Ritz pairs and define $S \in \mathbb{R}^{N\times k}$ as

$$S = V_l Y_{l,k} \tag{2.20}$$

with $Y_{l,k} \in \mathbb{R}^{l\times k}$ as $Y_{l,k} = [y_1, \cdots, y_k]$. We note that S has orthonormal columns, i.e. $S^TS = I_k$. Moreover, denoting $\Theta_k = diag(\theta_1, \cdots, \theta_k)$, a direct calculation leads to $S^TAS = \Theta_k$.

Before analyzing the Ritz limited memory preconditioner, we show two results concerning the spectral error associated to the Ritz vectors. First, Proposition 2.2.10 illustrates that these vectors are already approximations of their projection onto $\mathcal{I}_\pm(A)$ (depending on the sign of the associated Ritz value), using the Lanczos approximation of sign(A) (2.17) from the Lanczos relation (2.18). We emphasize that this approach is slightly different from the one described in Section 2.2.5: we use here the Lanczos approximation obtained from the solution of $Ax_1 = b_1$ (i.e. with initial vector $b_1/\|b_1\|$), instead of choosing as initial vector the vector f for which we want to compute $sign(A)f$. In other words, we apply a general Lanczos approximation of sign(A) instead of an approximation of the matrix-vector product sign$(A)f$. Secondly, Lemma 2.2.11 proves that $\mathcal{R}(S)$ is an invariant subspace of a matrix different from A.

Proposition 2.2.10. *Assume that l iterations of the Lanczos method have been performed so that the Lanczos relation (2.18) holds. Let s be a column of S in* (2.20)*, and θ the associated Ritz value. Using the Lanczos approximation of* sign(A) (2.17) *from the Lanczos relation* (2.18)*, we obtain*

$$\bar{s}_l = \text{sign}(\theta)s.$$

If $\theta > 0$, the associated approximate projection onto $\mathcal{I}_+(A)$ gives

$$\frac{s + \bar{s}_l}{2} = s.$$

If $\theta < 0$, the associated approximate projection onto $\mathcal{I}_-(A)$ gives

$$\frac{s - \bar{s}_l}{2} = s.$$

Proof. s being of column of S in (2.20), $s = V_l y$ where (y,θ) is an eigenpair of T_l. The

Lanczos approximation of $\text{sign}(A)s$ from the Lanczos relation (2.18) leads to

$$\begin{aligned}\bar{s}_l &= V_l \text{sign}(T_l) V_l^T s \\ &= V_l \text{sign}(T_l) V_l^T V_l y \\ &= V_l (T_l^2)^{-\frac{1}{2}} T_l y \\ &= V_l \frac{\theta}{|\theta|} y \\ &= \text{sign}(\theta) s.\end{aligned}$$

We recall that the exact projection of s onto $\mathcal{I}_{\pm}(A)$ is given by

$$P_{\pm}(A)s = \frac{(s \pm \text{sign}(A)s)}{2}.$$

Replacing $\text{sign}(A)s$ by $\bar{s}_l = \text{sign}(\theta)s$ in this relation completes the proof.

□

Lemma 2.2.11. *Assume that l iterations of the Lanczos method have been performed so that the Lanczos relation (2.18) holds. Let define the symmetric matrix $\Delta A \in \mathbb{R}^{N\times N}$ as*

$$\Delta A = -v_{l+1}(t_{l+1,l} e_l^T) V_l^T - V_l (t_{l+1,l} e_l) v_{l+1}^T. \tag{2.21}$$

Assume that $S \in \mathbb{R}^{N\times k}$ has been defined as in relation (2.20). Then $\mathcal{R}(S)$ is an invariant subspace of $(A + \Delta A)$ and

$$(A + \Delta A)S = S\Theta_k.$$

Proof. A simple calculation gives

$$(A + \Delta A)V_l = V_l T_l.$$

Postmultiplying by $Y_{l,k}$ leads to

$$(A + \Delta A)S = S\Theta_k,$$

with $\Theta_k = diag(\theta_1, \cdots, \theta_k)$. □

2.2.5.2 Characterization of the Ritz limited memory preconditioner

According to relation (2.7) of Theorem 2.2.2, we need to analyze $(P_{\mathcal{S}^\perp, A\mathcal{S}} A)|_{\mathcal{S}^\perp}$ to characterize the Ritz-LMP$_{\pm}$. This is detailed next in Theorem 2.2.12.

Theorem 2.2.12. *Let A be a symmetric indefinite matrix of order N and ΔA be given by (2.21) in Lemma 2.2.11. Assume that $S \in \mathbb{R}^{N\times k}$ has been defined as in relation (2.20). Then*

$$\|(P_{\mathcal{S}^\perp,(A+\Delta A)\mathcal{S}}A - P_{\mathcal{S}^\perp,A\mathcal{S}}A)|_{\mathcal{S}^\perp}\|_2 = t_{l+1,l}^2 \,|\sum_{i=1}^{k}\frac{y_{l,i}^2}{\theta_i}|,$$

with $y_{l,i} = e_l^T y_i$.

Proof. We consider the oblique projection $P_{\mathcal{S}^\perp,(A+\Delta A)\mathcal{S}}$ onto $\mathcal{S}^\perp$ along $(A+\Delta A)\mathcal{S}$ defined as

$$P_{\mathcal{S}^\perp,(A+\Delta A)\mathcal{S}} = I_N - (A+\Delta A)S(S^T(A+\Delta A)S)^{-1}S^T.$$

First, since $(S^T(A+\Delta A)S)^{-1} = \Theta_k^{-1}$, we note that

$$P_{\mathcal{S}^\perp,(A+\Delta A)\mathcal{S}} = I_N - SS^T = P_{\mathcal{S}^\perp}.$$

Furthermore, we have also $(S^TAS)^{-1} = \Theta_k^{-1}$ and $P_{\mathcal{S}^\perp,(A+\Delta A)\mathcal{S}}A$ can be written as

$$\begin{aligned} P_{\mathcal{S}^\perp,(A+\Delta A)\mathcal{S}}A &= (I_N - (A+\Delta A)S\Theta_k^{-1}S^T)A, \\ &= P_{\mathcal{S}^\perp,A\mathcal{S}}A - \Delta AS\Theta_k^{-1}S^TA, \\ &= P_{\mathcal{S}^\perp,A\mathcal{S}}A - \Delta AS(S^T - \Theta_k^{-1}S^T\Delta A). \end{aligned}$$

We note that $S^T\Delta AS = 0_k$ since $[V_l, v_{l+1}]$ is an orthonormal basis. Hence, $P_{\mathcal{S}^\perp}(\Delta AS) = \Delta AS$ which leads to

$$(P_{\mathcal{S}^\perp,(A+\Delta A)\mathcal{S}}A - P_{\mathcal{S}^\perp,A\mathcal{S}}A)|_{\mathcal{S}^\perp} = \Delta AS\Theta_k^{-1}S^T\Delta A. \tag{2.22}$$

Relation (2.21) yields

$$\Delta AS\Theta_k^{-1}S^T\Delta A = t_{l+1,l}^2(e_l^TY_{l,k}\Theta_k^{-1}Y_{l,k}^Te_l)v_{l+1}v_{l+1}^T.$$

Using the relations $\|uv^T\|_2 = \|u\|_2\|v\|_2$ and $\|v_{l+1}\|_2 = 1$ yields

$$\|\Delta AS\Theta_k^{-1}S^T\Delta A\|_2 = t_{l+1,l}^2 \,|\sum_{i=1}^{k}\frac{y_{l,i}^2}{\theta_i}|,$$

which completes the proof. □

Relation (2.22) reveals that

$$(P_{\mathcal{S}^\perp,A\mathcal{S}}A)|_{\mathcal{S}^\perp} = (P_{\mathcal{S}^\perp,(A+\Delta A)\mathcal{S}}A)|_{\mathcal{S}^\perp} - \tau v_{l+1}v_{l+1}^T, \tag{2.23}$$

with the scalar quantity τ defined by $\tau = t_{l+1,l}^2 e_l^T Y_{l,k}\Theta_k^{-1}Y_{l,k}^T e_l$. This shows that $(P_{\mathcal{S}^\perp,A\mathcal{S}}A)|_{\mathcal{S}^\perp}$ is equal to $(P_{\mathcal{S}^\perp}A)|_{\mathcal{S}^\perp}$ perturbed by a rank-one matrix. Hence, the application of Theorem 8.1.8 [43] shows the existence of nonnegative scalar quantities m_i such that

$$\lambda_i((P_{\mathcal{S}^\perp,A\mathcal{S}}A)|_{\mathcal{S}^\perp}) = \lambda_i((P_{\mathcal{S}^\perp}A)|_{\mathcal{S}^\perp})) + m_i\tau, \quad i = 1, \cdots, N-k, \tag{2.24}$$

with $m_1 + \cdots + m_{N-k} = 1$. If the columns of $Z_\perp \in \mathbb{R}^{N\times(N-k)}$ form an orthonormal basis for $\mathcal{S}^\perp$, the spectrum of $P_{\mathcal{S}^\perp}A|_{\mathcal{S}^\perp}$ is then given by

$$\Lambda(P_{\mathcal{S}^\perp}A|_{\mathcal{S}^\perp}) = \Lambda(Z_\perp^T A Z_\perp). \tag{2.25}$$

Relations (2.24) and (2.25) yield a simple characterization of the Ritz-LMP preconditioner which reveals that the scalar τ is an important quantity to monitor numerically. Indeed, when $|\tau|$ is small, $(P_{\mathcal{S}^\perp,A\mathcal{S}}A)|_{\mathcal{S}^\perp}$ is spectrally close to $(P_{\mathcal{S}^\perp}A)|_{\mathcal{S}^\perp}$.

2.3 Implementation considerations

In this section, we present possible implementations of the LMP$_\pm$, either in the general case or with Ritz vectors as columns of S in Definition 2.2.1. A discussion about the use of harmonic Ritz vectors is also proposed. We detail here the related computational cost and specify the memory requirements.

2.3.1 Computational cost and memory requirements for a general matrix S

First, we consider a matrix S in (2.6) such that S^TAS is nonsingular. We need to consider $(S^TAS)^{-1}$, but according to Proposition 2.2.1, we can replace S by Z with columns of Z spanning the same subspace $\mathcal{S}$. In [46], a Gram-Schmidt process is used to compute Z such that $Z^TAZ = I_k$ when A is symmetric positive definite. In the indefinite case, A does not define any longer an inner product and we propose a slightly different method, building Z which satisfies $(Z^TAZ)^{-1} = \Sigma = \mathrm{diag}(\sigma_1, ..., \sigma_k)$. Then, denoting $Y = AZ\Sigma$, the LMP$_\pm$ can be written as

$$H = (I_N - ZY^T)(I_N - YZ^T) + Z\Sigma Z^T. \tag{2.26}$$

This orthogonalization is detailed in Algorithm 5, setting $Z_i = [z_1, ..., z_i]$ and $Y_i = [y_1, ..., y_i]$, for $i = 1, ..., k$. In terms of computational cost, this process requires k matrix-vector products by A and $3N + \sum_{i=1}^{k-1}(4iN + 4N) = (2k^2 + 2k - 1)N$ additional floating point operations. We further note that the aim is not to compute H but just to compute and store Z, Y and Σ. The additional storage is then approximatively of $2k$ vectors of length N. Finally, it is possible to apply H on a vector in $8kN + k \sim 8kN$ floating point operations (see Algorithm 6).

Algorithm 5 Compute Z, Y and Σ such that $H = (I_N - ZY^T)(I_N - YZ^T) + Z\Sigma Z^T$

1:	$z_1 = s_1$, $y_1 = As_1$	(one product by A)
2:	$\sigma_1 = \frac{1}{z_1^T y_1}$	($2N$ flops)
3:	$y_1 = y_1\sigma_1$	(N flops)
4:	**for** $i = 1$ to $k - 1$ **do**	
5:	$f = Y_i^T s_{i+1}$	($2iN$ flops)
6:	$z_{i+1} = s_{i+1} - Z_i f$	($2iN + N$ flops)
7:	$y_{i+1} = Az_{i+1}$	(one product by A)
8:	$\sigma_{i+1} = \frac{1}{z_{i+1}^T y_{i+1}}$	($2N$ flops)
9:	$y_{i+1} = y_{i+1}\sigma_{i+1}$	(N flops)
10:	**end for**	

Algorithm 6 Application of the limited memory preconditioner: $r = Hx$

1:	$f = Z^T x$	($2kN$ flops)
2:	$\bar{f} = \Sigma f$	(k flops)
3:	$\bar{r} = (x - Yf)$	($2kN$ flops)
4:	$r = \bar{r} - Z(Y^T\bar{r} - \bar{f})$	($4kN$ flops)

2.3.2 Computational cost and memory requirements of the Ritz-LMP$_\pm$

We consider here the Ritz-LMP$_\pm$, i.e. the k columns of S in (2.6) are selected as Ritz vectors. By exploiting the Lanczos relation (2.18), a significant reduction of the complexity and the memory requirements can be obtained. Indeed, Theorem 4.3 in [46] shows that the Ritz-LMP$_\pm$ H reads as

$$H = I_N + S(\Theta_k^{-1} - I_k)S^T - S\omega v_{l+1}^T - v_{l+1}\omega^T S^T + S\omega\omega^T S^T, \tag{2.27}$$

where the components of $\omega = (\omega_1, \cdots, \omega_k)^T$ are defined as

$$\omega_i = \frac{t_{l+1,l} y_{l,i}}{\theta_i}, (i = 1, \cdots, k). \tag{2.28}$$

Later we define the vector $s_\omega \in \mathbb{R}^N$ as $s_\omega = S\omega$. Hence, the application of the preconditioner H on a vector of length N costs $(4k+9)N$ floating point operations as detailed in Algorithm 7. In addition, the storage requirements related to the Ritz-LMP$_\pm$ variant consist of $k+2$ vectors of length N (v_{l+1}, s_ω and $S \in \mathbb{R}^{N\times k}$) together with k scalars (corresponding to the Ritz values).

Algorithm 7 Application of the Ritz limited memory preconditioner: $r = Hx$

1: $\alpha_1 = s_\omega^T x$ (costs $2N$ flops)
2: $\alpha_2 = -\alpha_1 + v_{l+1}^T x$ (costs $2N$ flops)
3: $z = S(\Theta_k^{-1} - I_k)S^T x$ (costs $4kN$ flops)
4: $r = x + z - \alpha_2 s_\omega - \alpha_1 v_{l+1}$. (costs $5N$ flops)

This technique is interesting compared to the general case. Indeed, we do not need to perform the orthogonalization process in Algorithm 5, and each application of H on a vector is cheaper in terms of floating point operations as soon as $k > 2$. Furthermore, the number of vectors to store is lower, also for $k > 2$. This technique for the Ritz-LMP$_\pm$ is essentially based on the A-orthogonality and the orthonormality of S, i.e.

$$S^T AS = \Theta_k \quad \text{and} \quad S^T S = I_k. \tag{2.29}$$

2.3.3 Case of harmonic Ritz vectors as columns of S

The harmonic Ritz vectors do not satisfy the relations (2.29) and we cannot adapt Algorithm 7 to this case. Moreover, contrary to the Ritz vectors which are defined from eigenvectors of the symmetric matrix T_l in (2.18), the harmonic Ritz vectors can be complex-valued. Indeed, as introduced in 1.2.3, they are of the form $s = V_l y$, where y is a solution of the real nonsymmetric eigenproblem

$$(T_l + |t_{l+1,l}|^2 T_l^{-T} e_l e_l^T)y = \theta y.$$

Nevertheless, if (y, θ) satisfies this equation, the conjugate pair $(\bar{y}, \bar{\theta})$ is also a solution. Since $span\{Re(y), Im(y)\} = span\{y, \bar{y}\}$, it is possible to define two columns of $S \in \mathbb{R}^{N\times k}$ selecting both vectors $V_l Re(y)$ and $V_l Im(y)$. This strategy adds a condition on the choice of S in this case: for an imposed value k, we may allow the method to select $k+1$ columns in S to ensure the inclusion of both vectors associated to conjugate harmonic Ritz values. Finally, we can use Algorithms 5 and 6 to apply the LMP$_\pm$. The cost per application of H is then more important than using Ritz vectors.

2.4 Applications to solid mechanics

Now, we illustrate the numerical behaviour of limited memory preconditioners on applications in solid mechanics that require the solution of symmetric saddle point linear systems, as described in Section 1.1. This preconditioning technique has been implemented in Code_Aster using the PETSc library (version 3.4.5). Some information about the implementation is given in Appendix A.

2.4.1 Sequence of preconditioned saddle point systems

As introduced in Section 1.1.3, we consider a sequence of linear systems of saddle point type

$$\mathcal{K}_i \; y_i = c_i \Longleftrightarrow \begin{pmatrix} G_i & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u_i \\ \lambda_i \end{pmatrix} = \begin{pmatrix} f_i \\ g_i \end{pmatrix}, \quad i = 1, \cdots, I, \tag{2.30}$$

where $m < n$, $B \in \mathbb{R}^{m \times n}$, $f_i \in \mathbb{R}^n$, $g_i \in \mathbb{R}^m$ and $G_i \in \mathbb{R}^{n \times n}$ are symmetric positive semidefinite stiffness matrices. We further assume that B is of full row rank ($rank(B) = m$) and that $\mathcal{N}(G_i) \cap \mathcal{N}(B) = \{0\}$, $\forall i \in \{1, \cdots, I\}$, which make sure the existence and uniqueness of the solution of each linear system in the sequence (see Theorem 1.3.1).

In general, Krylov subspace methods are only feasible in combination with a preconditioner when considering large-scale problems [90]. Before involving the LMP$_\pm$, we consider the action of a first-level preconditioner: we use here the specific block diagonal symmetric positive definite preconditioner based on the augmented Lagrangian method and introduced in (1.23). It is computed from the first saddle point matrix $\mathcal{K}_1$ in the sequence 2.30 and we recall here its definition:

$$\mathcal{M}_d = \begin{pmatrix} G_1 + \gamma B^T B & 0 \\ 0 & \frac{1}{\gamma} I_m \end{pmatrix}, \quad \gamma > 0. \tag{2.31}$$

Since inverting exactly $G_1 + \gamma B^T B$ is too demanding in terms of both computational operations and memory requirements for large-scale problems, we consider a factorized approximate preconditioner of the form $\mathcal{M}_d \approx \mathcal{L}\mathcal{L}^T$ based on the incomplete Cholesky factorization of $G_1 + \gamma B^T B$ written as $G_1 + \gamma B^T B \approx LL^T$ [90]. We deduce the final formulation of the symmetric preconditioned linear system $\mathcal{A}_i \; x_i = b_i$ denoted as

$$\mathcal{A}_i \; x_i = b_i \Longleftrightarrow \begin{pmatrix} L^{-1} & 0 \\ 0 & \sqrt{\gamma} I_m \end{pmatrix} \begin{pmatrix} G_i & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} L^{-T} & 0 \\ 0 & \sqrt{\gamma} I_m \end{pmatrix} \begin{pmatrix} w_i \\ z_i \end{pmatrix} = \begin{pmatrix} L^{-1} & 0 \\ 0 & \sqrt{\gamma} I_m \end{pmatrix} \begin{pmatrix} f_i \\ g_i \end{pmatrix}. \tag{2.32}$$

We emphasize that the property of symmetric definite positiveness of the first-level preconditioner is necessary to keep the symmetry of $\mathcal{A}_i$.

Limited memory preconditioners combined with Krylov subspace methods will be used to solve the sequence of linear systems (2.32) approximately. As mentioned in Section 2.2.5, we extract k approximations of eigenvectors known as Ritz vectors (unless otherwise stated) when solving the first linear system in this sequence to deduce $S \in \mathbb{R}^{N\times k}$. We note that S is used in the whole sequence, even in the case of changing matrices (as in Sections 2.4.4 and 2.4.5). Hence, with this choice, the limited memory preconditioner H is then defined once for all as

$$H = (I_{n+m} - S(S^T\mathcal{A}_1S)^{-1}S^T\mathcal{A}_1)(I_{n+m} - \mathcal{A}_1S(S^T\mathcal{A}_1S)^{-1}S^T) + S(S^T\mathcal{A}_1S)^{-1}S^T. \quad (2.33)$$

In all the applications considered, we always select the Ritz vectors corresponding to the smallest in modulus Ritz values. Since positive or negative Ritz values can occur in practice, H is a symmetric indefinite preconditioner due to Theorem 2.2.4. In particular, we cannot use MINRES, a method of choice to solve symmetric systems, which necessits a symmetric positive definite preconditioner [99]. Hence we use the symmetric indefinite matrix H as a right preconditioner of GMRES(30) for the approximate solution of the remaining linear systems in the sequence (2.32). To summarize, this Krylov subspace method is used to solve

$$\mathcal{A}_1x_1 = b_1 \quad \text{and} \quad \begin{cases} \mathcal{A}_iH\tilde{x}_i = b_i \\ x_i = H\tilde{x}_i \end{cases} \qquad i = 2, \cdots, I.$$

We note that the Ritz vectors are computed using the Lanczos relation obtained from the last complete cycle of GMRES(30), when solving $\mathcal{A}_1x_1 = b_1$. Furthermore, a zero initial guess is always chosen and the iterative method is stopped when the Euclidean norm of the residual normalized by the Euclidean norm of the right-hand side satisfies the following relation

$$\frac{||b_i - \mathcal{A}_ix_i^k||_2}{||b_i||_2} \leq 10^{-8}. \quad (2.34)$$

We remark that the stopping criterion is fixed for all linear systems of the sequence (2.32), which means that we do not use an inexact Newton method (see, e.g., [111, 105]).

The numerical results have been obtained on the Aster5 cluster, a IBM IDATAPLEX computer located at EDF R&D Data Center (each node of Aster5 is equipped with 2

Intel Xeon E5 − 2600, each running 12 cores at 2.7 Ghz). Physical memory available on a given node (24 cores) of Aster5 ranges from 64 GB to 1 TB. This code was compiled by the Intel compiler suite with the best optimization options and linked with the Intel MKL BLAS and LAPACK subroutines. Both iteration counts and computational times will be reported. Our main interest is to analyze the efficiency of the limited memory preconditioner for the solution of the sequence of saddle point systems where both the matrices and the right-hand sides may change. A small-scale problem is considered first, while two large-scale real-life industrial configurations will be analyzed later in Sections 2.4.3 and 2.4.5. Problems with multiple right-hand sides only are considered in Sections 2.4.2 and 2.4.3, while sequences of linear systems with changing matrices are addressed in Sections 2.4.4 and 2.4.5.

2.4.2 Mechanical bearing (linear case)

We first focus on a linear problem in solid mechanics related to the computation of the displacement of a mechanical bearing. In this experiment, the bearing is subject to an external pressure on its left part, while embedded on the right part. The computational mesh is shown in Figure 2.5.

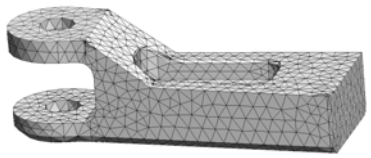

Figure 2.5: Mesh of the mechanical bearing.

The matrix B in (2.32) corresponds to the dualization of the single freedom constraints from the clamping of the structure. As described in Section 1.3.2.2, the coefficient γ in the augmented Lagrangian preconditioner is chosen equal to the scaling coefficient α immediately available in Code_Aster. Here, $\gamma = 2.9027 \times 10^8$. The moderate dimension of the problem ($n = 7305$, $m = 228$, $N = 7533$) allows us to compute the spectrum of $\mathcal{A}$ (using the SLEPc library [4, 53], version 3.5.2), as illustrated in Figure 2.6. It exhibits clusters at -1 and 1 (in agreement with the theory, see Theorem 1.3.2) and eigenvalues close to 0.

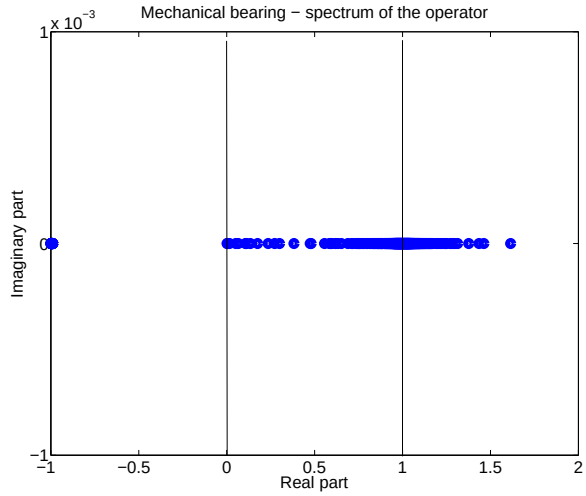

Figure 2.6: Mechanical bearing: spectrum of $\mathcal{A}$.

In this linear case, the sequence (2.32) is reduced to one linear system denoted $\mathcal{A}x = b$. We have solved it once, without LMP$_\pm$, and we have recovered some information to define H. Then we have solved the system a second time with different LMP$_\pm$. This small-case model example give us attractive information. First, we investigate the behaviour of the limited memory preconditioner with S based either on Ritz vectors (Ritz-LMP$_\pm$) or harmonic Ritz vectors, both related to smallest in modulus (harmonic) Ritz values. Figure 2.7 shows the convergence history of GMRES(30) for both variants of LMP$_\pm$ with increasing of values of k ($k = 5, 20, 30$, respectively). We note that the use of the limited memory preconditioner leads to a significant decrease in terms of numbers of iterations. The combination of the augmented Lagrangian preconditioner with the limited memory preconditioner is thus successful on this application. In addition we note that the use of Ritz and harmonic Ritz vectors in S leads to similar convergence histories. This strengthens our belief in selecting Ritz vectors to define S, knowing that the related computational cost is lower than selecting harmonic Ritz vectors (see Section 7). Although not illustrated, this type of behaviour arises in all numerical experiments in this chapter.

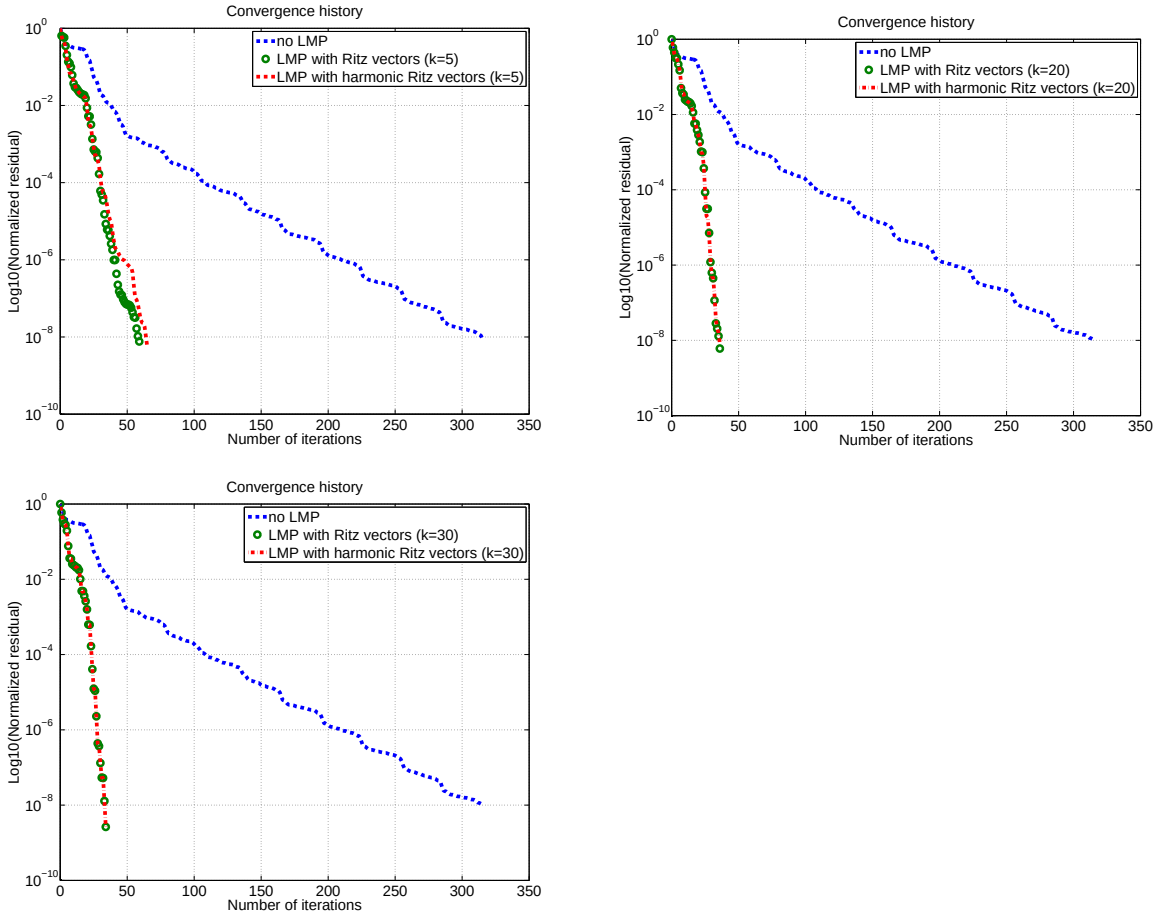

Figure 2.7: Mechanical bearing (linear case): convergence history of GMRES(30). Three preconditioning methods are compared: no second level preconditioning, limited memory preconditioners based on $k = 5$, 20 or 30 Ritz vectors and harmonic Ritz vectors, respectively.

Besides the spectrum of $\mathcal{A}$, we have been able to compute the matrix sign function of $\mathcal{A}$ using the eigenpairs obtained with SLEPc (see [55]). Thus we can analyse the effect of the limited memory preconditioner with S based either on exact eigenvectors (spectral-LMP$_\pm$), Ritz vectors (Ritz-LMP$_\pm$) or projected Ritz vectors as introduced in Section 2.2.4. They are all related to smallest in modulus eigenvalues or Ritz values. We note in Figure 2.8 that using eigenvectors of $\mathcal{A}$ for S leads to the minimal number of iterations in all situations. More interestingly, we also note that using Ritz or projected Ritz vectors for S leads to a very similar convergence behaviour on this model example. This fact is in agreement with the theory presented in Section 2.2.5 and suggests that in practice it is sufficient to consider Ritz vectors only. We further note that the value of $|\tau|$

given in relation (2.23) is equal to 0.0335, 0.075 and 0.079 for $k = 5, 20, 30$, respectively. We will thus consider only the Ritz-LMP$_\pm$ variant, based on Ritz vectors related to Ritz values of smallest modulus in the next sections of this chapter.

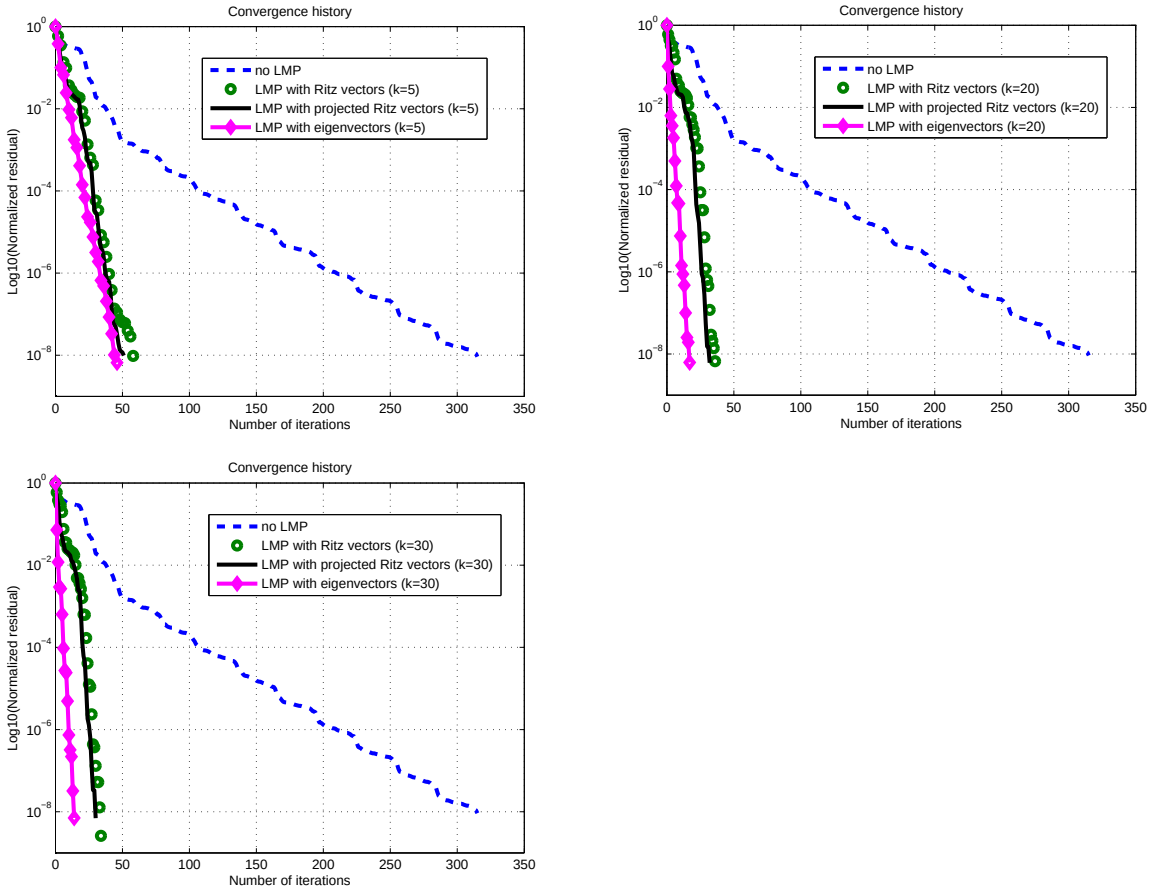

Figure 2.8: Mechanical bearing (linear case): convergence history of GMRES(30). Four preconditioning methods are compared: no second level preconditioning, limited memory preconditioners based on $k = 5$, 20 or 30 Ritz vectors, projected Ritz vectors and exact eigenvectors, respectively.

2.4.3 Containment building of a nuclear reactor

In this section, we investigate the mechanical properties of the containment building of a nuclear reactor of a Pressurized Water Reactor power plant. This building protects both the reactor from external aggressions and the environment if an internal accident occurs. Robust and accurate numerical simulations are thus required for both design and safety analysis. We consider an advanced mechanical modeling that takes into account

numerous prestressing tendons, whose role is to improve the global resistivity of the structure (see Figure 2.9). The containment building is subject to gravity and to an internal pressure. The whole loading is gradually applied into 4 successive steps. Each pitch of loading then corresponds to a specific linear system in the sequence, where only the right-hand side has changed (i.e. $\mathcal{A}_1 = ... = \mathcal{A}_4$). The introduction of Lagrange multipliers stems from the imposition of kinematic relations modeling perfect adhesion between the prestressing tendons and the concrete [70] and to the dualization of the essential boundary conditions. In this setting, B admits either five or one nonzero entries per row, respectively. This study is known to be complex for different reasons. First, from a mechanical point of view, the modeling is rather advanced with a mixing of three-dimensional elements for the concrete, two-dimensional elements for the metal shell placed on the intern wall of the building (to insure the sealing if an accidental leak occurs), and of one-dimensional elements for the tendons. Moreover, since the prestressing tendons are attached to the concrete thanks to dualized linear relations, the number of Lagrange multipliers is really important ($m = 158928$ for a global size of $N = 442725$). The number of nonzero entries of G_1 and $G_1 + \gamma B^T B$ is 7079238 and 8343685, respectively. Secondly, the occurrence of a large number of prestressing tendons (more than 600 here) induces a nullspace of large dimension for the stiffness matrix. Actually, it is known that this dimension is related to the number of rigid body motions of the subbodies of materials contained within the finite element mesh [76]. This numerical study is thus challenging and serves as a relevant realistic test case in solid mechanics to investigate the efficiency of preconditioners for Krylov subspace methods.

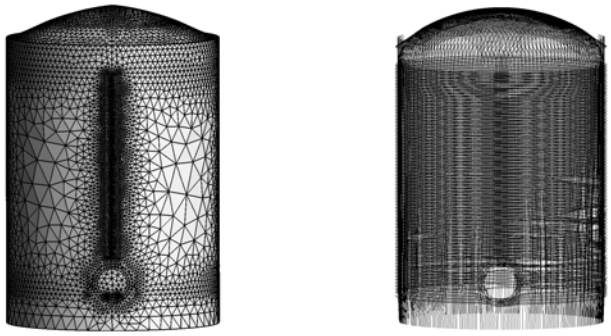

Figure 2.9: Containment building: three-dimensional mesh (left part) and location of the prestressing tendons on the surface (right part).

In this experiment, we set γ to $2,4684 \times 10^{11}$ and consider a level of fill equal to 8 in the incomplete Cholesky factorization of the $(1,1)$ block of $\mathcal{M}_d$. Actually, with a lower level of fill the preconditioned Krylov subspace method can hardly converge. However,

even with this value of fill, the required memory is around 7 Go, while state-of-the-art sparse direct solvers require at least 10 Go for the complete factorization of the $(1,1)$ block of $\mathcal{M}_d$.

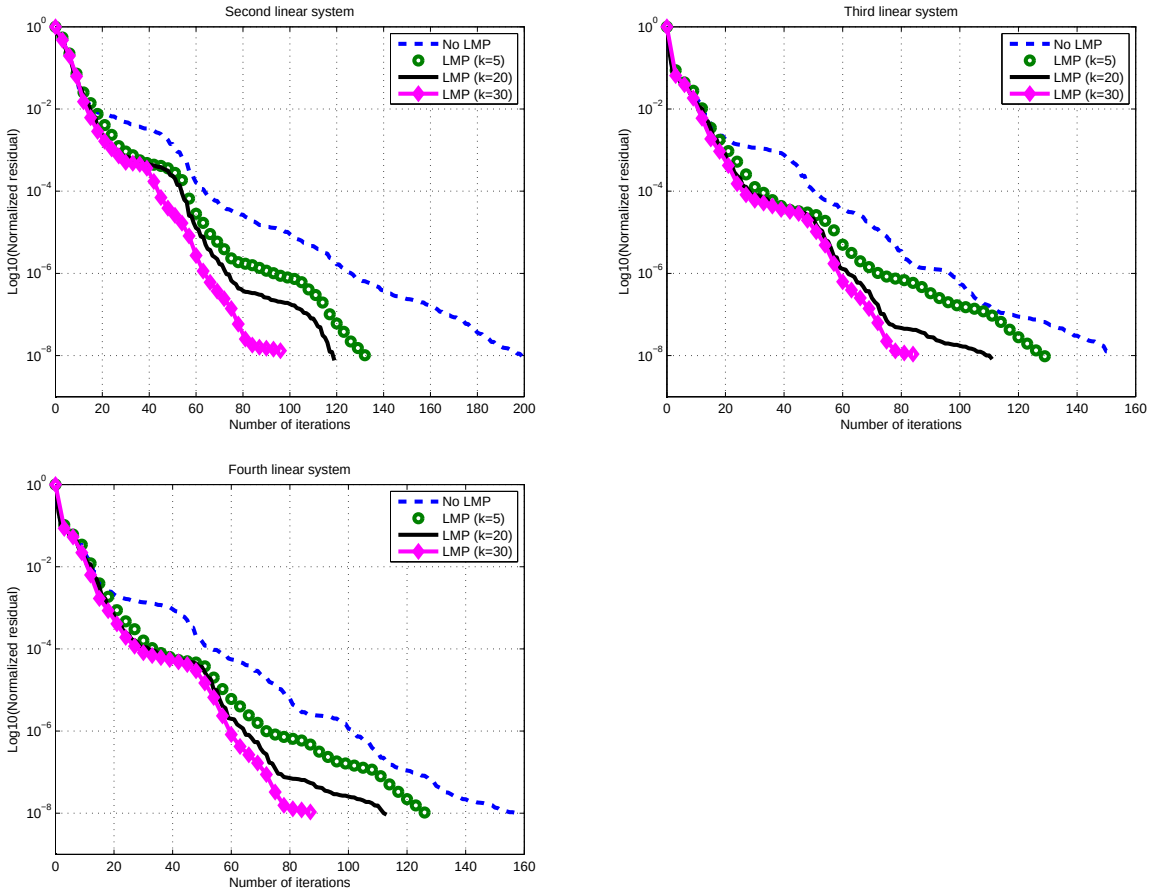

Figure 2.10: Containment building: convergence history of preconditioned GMRES(30) for the last three linear systems in the sequence. Case of limited memory preconditioners with $k = 5$, 20 or 30 Ritz vectors.

Figure 2.10 shows the evolution of the Euclidean norm of the relative residual for the last three linear systems in the sequence ($i = 2, 3, 4$). In this experiment, we consider limited memory preconditioners with a varying number of Ritz vectors ($k = 5$, 20, 30, respectively). Whatever the linear system considered in the sequence, the smallest number of iterations is obtained when selecting a large value of Ritz vectors ($k = 30$). In addition, we show in Table 2.1 the cumulative iteration count over the last three linear systems, the CPU time and the memory requirements provided by PETSc, respectively. We note that selecting S based on $k = 30$ Ritz vectors leads to a decrease of 47% in

terms of cumulated iterations and to a decrease of 41% in terms of CPU time. This satisfactory result comes at a price of a very moderate increase in memory requirements (3%), since the limited memory preconditioner only needs the storage of $(k+2)$ vectors of size N as detailed in Section 2.3.2.

	No LMP$_\pm$	LMP$_\pm$, $k=5$	LMP$_\pm$, $k=20$	LMP$_\pm$, $k=30$
Total iteration count	509	389	343	272
Iteration count decrease (%)	×	24	33	**47**
CPU time (sec)	315	254	224	186
CPU time decrease (%)	×	19	29	**41**
Memory (Mo)	6686	6722	6823	6891
Memory increase (%)	×	**0.5**	2	3

Table 2.1: Containment building: cumulative iteration count for the last three linear systems in the sequence, CPU time and memory requirements for different limited memory preconditioners. Case of $k = 5$, 20 or 30 Ritz vectors.

The Ritz-LMP$_\pm$ has proved to be efficient in these first numerical experiments in terms of both preconditioner applications and computations time. In both cases, the matrix is fixed and the next numerical experiments aim at illustrating the behaviour of GMRES for a sequence of linear systems with slowly varying left-hand sides.

2.4.4 Mechanical bearing (nonlinear case)

Before considering a large-scale nonlinear test case, we study once again the mechanical bearing problem described in Section 2.4.2. Here we consider a nonlinear constitutive law of the material (elastoplastic law of Von Mises with isotropic hardening), and we obtain from the Newton's method a sequence of four linear systems of the form (2.32) with multiple right and left-hand sides. In this small-scale case, we have been able to compute the difference between the successive original matrices $\mathcal{K}_i$ and preconditioned matrices $\mathcal{A}_i$, respectively. They are defined, for $i = 1, 2, 3$, as

$$d^i_{\mathcal{K}} = \frac{\|\mathcal{K}_{i+1} - \mathcal{K}_i\|_F}{\|\mathcal{K}_i\|_F} = \frac{\|G_{i+1} - G_i\|_F}{\|G_i\|_F}$$

and

$$d^i_{\mathcal{A}} = \frac{\|\mathcal{A}_{i+1} - \mathcal{A}_i\|_F}{\|\mathcal{A}_i\|_F} = \frac{\|L^{-1}(G_{i+1} - G_i)L^{-T}\|_F}{\|L^{-1}G_iL^{-T}\|_F},$$

where $\|.\|_F$ corresponds to the Frobenius norm. Here $d^1_{\mathcal{K}} = 4.758 \times 10^{-2}$, $d^2_{\mathcal{K}} = 8.755 \times 10^{-3}$ and $d^3_{\mathcal{K}} = 2.216 \times 10^{-4}$. The differences concerning the preconditioned matrices

are similar, with $d_{\mathcal{A}}^1 = 1.508 \times 10^{-2}$, $d_{\mathcal{A}}^2 = 5.935 \times 10^{-3}$ and $d_{\mathcal{A}}^3 = 2.774 \times 10^{-4}$.

As in the linear case, the use of the Ritz-LMP$_\pm$ leads to a significant decrease of GMRES(30) iteration count for the last three linear systems of the sequence (see Figure 2.11). In Table 2.2, we remark that this gain can reach up to 70% for $k = 30$, while a decrease in terms of CPU time of 66% is obtained (always with a negligible memory overcost).

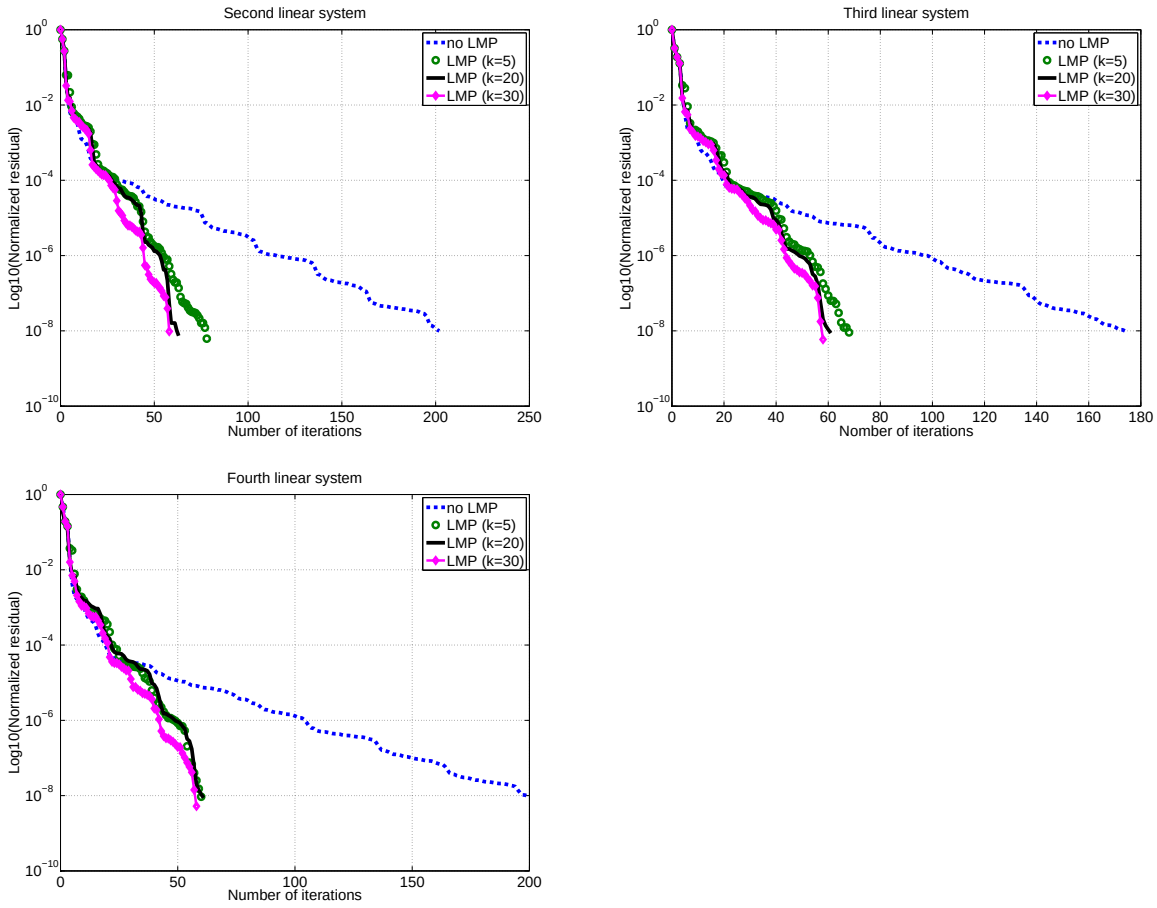

Figure 2.11: Mechanical bearing (nonlinear case): convergence history of preconditioned GMRES(30) for the last three linear systems in the sequence. Case of limited memory preconditioners with $k = 5$, 20 or 30 Ritz vectors.

	No LMP$_\pm$	LMP$_\pm$, $k = 5$	LMP$_\pm$, $k = 20$	LMP$_\pm$, $k = 30$
Total iteration count	575	206	182	174
Iteration count decrease (%)	×	64	68	**70**
CPU time (sec)	1.96	0.66	0.69	0.69
CPU time decrease (%)	×	65	65	**66**
Memory (Mo)	411	412	414	416
Memory increase (%)	×	**0.2**	0.7	1.2

Table 2.2: Mechanical bearing (nonlinear case): cumulative iteration count for the last three linear systems in the sequence, CPU time and memory requirements for different limited memory preconditioners. Case of $k = 5$, 20 or 30 Ritz vectors.

2.4.5 Polycrystalline aggregate

The polycrystalline aggregate problem is especially used as an homogenization method to obtain macroscopic constitutive laws of a material from microscopic considerations only. In this framework, numerical simulations are performed at a mesoscopic scale in a simple geometry (a cube named representative elementary volume). One thousand points are randomly distributed in this cube and Voronoi cells are created using perpendicular bisector planes. Each cell then represents a grain which has its own constitutive law. The cells are finally discretized with tetrahedra leading to a global three-dimensional unstructured mesh (see Figure 2.12 for an illustration).

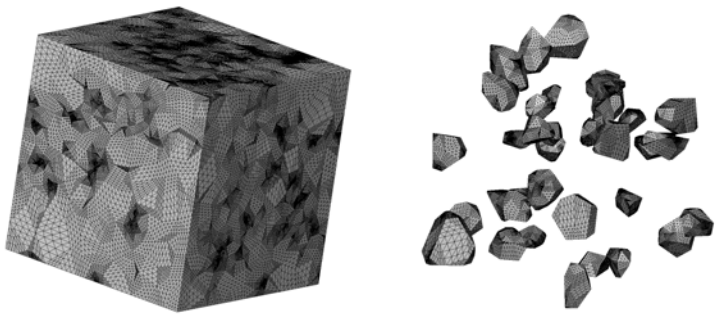

Figure 2.12: Polycrystalline aggregate: unstructured mesh of the representative elementary volume (left part) and detailed view of some grains of the polycrystalline aggregate (right part).

We impose a traction loading on a given face of the cube and specify zero displacement boundary conditions on the other faces as shown in Figure 2.13.

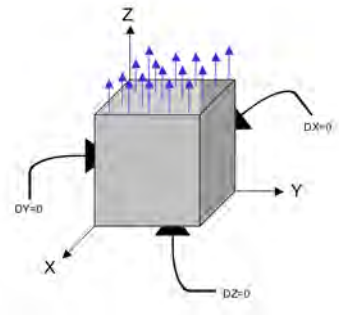

Figure 2.13: Polycrystalline aggregate: sketch of the boundary conditions.

All these boundary conditions are dualized, leading to a sequence of saddle point systems of the form (2.30), with slowly varying left-hand sides. To evaluate the numerical performance of the limited memory preconditioners, we aim at performing these simulations on different meshes. Thus a coarse mesh ($N = 56561$ with $n = 54567$ and $m = 1994$), an intermediate mesh ($N = 425222$ with $n = 417525$ and $m = 7697$) and a fine mesh ($N = 3298601$ with $n = 3268359$ and $m = 30242$) are considered in this study. We note that the proportion of Lagrange multipliers is always less than 1% and that the simulation on the finest mesh is considered as a real computational challenge in practice. Newton's method is employed because of the nonlinearity of the constitutive law of the structure. As expected, it is found that the total number of Newton's iterations is mesh-dependent (7, 9, 14 iterations are required on the coarse, intermediate and fine mesh, respectively). Finally, we set γ to 1.052×10^5, 7.6101×10^4 and 7.3267×10^4 on the coarse, intermediate and fine mesh, respectively. Similarly, we set the level of fill for the incomplete Cholesky factorization of the $(1,1)$ block of $\mathcal{M}_1$ to 4 in the three cases.

Table 2.3 collects the results for the three different simulations. Whatever the level of mesh refinement, we observe that the use of the limited memory preconditioner leads to a significant decrease both in terms of cumulative number of iterations over the whole Newton's sequence and of computational operations. A decrease of 17.2% in terms of computational time at a price of a low memory increase (only 0.2%) is indeed obtained on the fine mesh calculation which is a rather satisfactory result. Larger gains are obtained for the simulations on the coarse and intermediate meshes. On this application, choosing 5 Ritz vectors leads to the best strategy in terms of floating point operation reduction. Considering a larger number of Ritz vectors reduces the cumulative number of iterations as shown in Figure 2.14. Nevertheless, this choice leads to a larger cost in terms of computational operations.

		No LMP$_\pm$	LMP$_\pm$, $k=5$	LMP$_\pm$, $k=20$	LMP$_\pm$, $k=30$
Coarse mesh	Total iteration count	354	235	227	222
	Iteration count decrease (%)	×	33.5	36	**37.5**
	CPU time (sec)	24.8	16.6	17.3	17.6
	CPU time decrease (%)	×	**33.1**	30.2	29.1
	Memory (Mo)	1137	1140	1146	1151
	Memory increase (%)	×	**0.2**	0.8	1.2
Interm. mesh	Total iteration count	1316	1033	1027	1019
	Iteration count decrease (%)	×	21.5	22	**22.5**
	CPU time (sec)	528	434	447	455
	CPU time decrease (%)	×	**17.9**	15.3	13.8
	Memory (Mo)	8286	8305	8358	8387
	Memory increase (%)	×	**0.2**	0.8	1.2
Fine mesh	Total iteration count	6002	4835	4651	4614
	Iteration count decrease (%)	×	20	22.5	**23**
	CPU time (sec)	19920	16683	16498	16682
	CPU time decrease (%)	×	**17.2**	16.3	16.3
	Memory (Mo)	65613	65787	66165	66416
	Memory increase (%)	×	**0.2**	0.8	1.2

Table 2.3: Polycrystalline aggregate: cumulative iteration count over the complete Newton's sequence, CPU time and memory requirements for different preconditioners. Results are given for three different levels of mesh refinement (coarse, intermediate and fine, respectively).

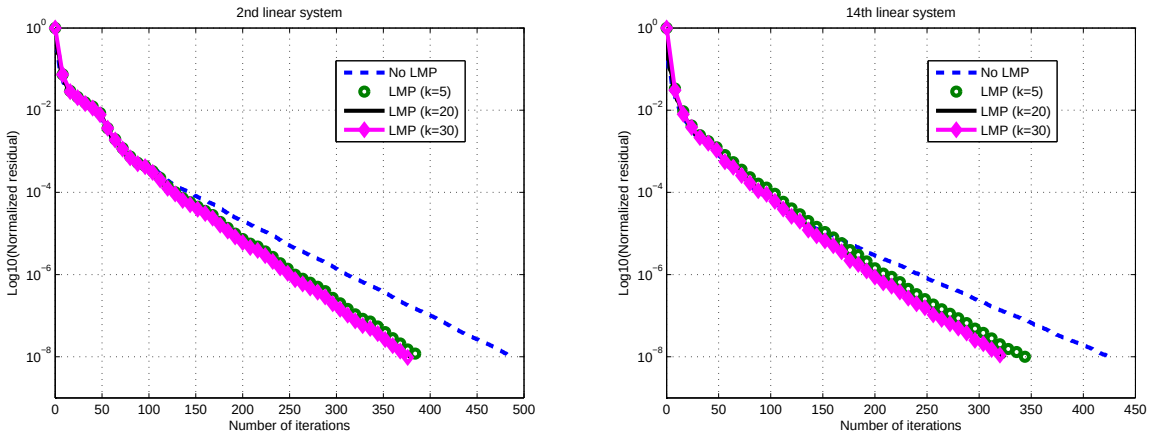

Figure 2.14: Polycrystalline aggregate (fine mesh calculation): GMRES(30) convergence history for different preconditioners at 2 different iterations of the Newton's method (2nd and 14th iterations).

2.5 Conclusions

We have proposed in this chapter a class of limited memory preconditioners for the solution of linear systems with symmetric indefinite matrices and multiple right-hand sides. This preconditioner based on limited memory quasi-Newton formulas can notably be used to improve an existing first-level symmetric definite positive preconditioner (applied symmetrically as a split preconditioner). In addition, this method is especially worth considering when the solution of a sequence of linear systems with slowly varying left-hand sides is considered.

We have derived a formula to characterize the spectrum of the preconditioned operator. We have shown that the eigenvalues of the preconditioned operator are real-valued (with at least k eigenvalues equal to 1). Furthermore, we have shown that the eigenvalues of the preconditioned matrix enjoy interlacing properties with respect to the eigenvalues of the original matrix provided that the k linearly independent vectors have been prior projected onto invariant subspaces associated with the eigenvalues of the original matrix. Then, we have studied the Ritz-LMP$_\pm$ variant, where Ritz information is used to determine the k vectors.

In practice, we have tested the Ritz-LMP$_\pm$ variant on solid mechanics problems, as introduced in Chapter 1, and this variant has proved to be efficient in terms of both preconditioner applications and computational operations. Numerical experiments have highlighted the relevance of the proposed preconditioner that leads to a significant decrease in terms of computational operations. A saving of up to 41% in terms of computational time is obtained with respect to the original method on one of the large-scale applications.

Although not reported in this chapter, the proposed limited memory preconditioner formula has been also implemented in a parallel distributed memory environment through both the Code_Aster and PETSc libraries. In practice, this straightforward extension allows us to consider selected large-scale industrial problems in a limited amount of computational time on a moderate number of cores. The scalability properties will be illustrated in Chapter 3, where the main focus is on the derivation of preconditioner update formulas in the case where the original preconditioned matrix is not symmetric. In addition, when the original matrix is symmetric, this extension also allows us to consider a broader class of first-level preconditioners (without the symmetric definite positiveness restriction). We further emphasize that a comparison with existing methods is performed in Chapter 3, where we also consider the LMP$_\pm$.

Chapter 3

Limited memory preconditioners for nonsymmetric systems

Motivated by the encouraging results obtained with the limited memory preconditioners proposed for symmetric indefinite systems in Chapter 2, we have also studied the nonsymmetric case. Although the solid mechanics problems treated in this manuscript lead to sequences of symmetric linear systems (with saddle point structure), the purpose of such an approach is to overcome the constraint related to the symmetric definite positiveness property of the first level preconditioner required in Chapter 2. Besides this situation, the algebraic method developed in this chapter can obviously be applied on any nonsingular linear system.

As in the symmetric indefinite case, we have considered update formulas issued from the numerical optimization literature to define efficient improvement preconditioning techniques. In this sense, Section 3.1 details how the proposed class of limited memory preconditioners for nonsymmetric linear systems (named LMP_{ns}) has been obtained, from both Broyden method and EN method (Eirola and Nevanlinna). This family is analysed in Section 3.2, where the main contribution concerns the spectral characterization of the preconditioned operator. Even if this class of limited memory preconditioners is new, to the best of our knowledge, the involved operators have already been used either in an abstract balancing preconditioner [33], or in deflated Krylov subspace methods [25, 39, 33]. Section 3.3.1 finally aims at positioning and comparing the LMP_{ns} with these existing techniques, where we expose our two main contributions. We emphasize this in a more general framework and include in this comparison the class of limited memory preconditioners proposed in Chapter 2. First, we show that the spectra of the respective associated operators are equal or at least very close in some sense. Secondly,

we compare the behaviour of GMRES combined with these methods and we particularly show that the abstract balancing preconditioners, equivalent to the limited memory preconditioners in the symmetric indefinite case, provide the same GMRES iterates than deflation under some assumptions. After discussing some implementation issues in Section 3.4, we illustrate in Section 3.5 the numerical efficiency of the limited memory preconditioner on possibly large-scale applications in solid mechanics, as introduced earlier in this manuscript. This preconditioning technique is also numerically compared with the deflated GMRES method.

3.1 Motivations and definition

This study is based on quasi-Newton methods intended to determine a zero of a function F, using approximations either of the Jacobian of F or of its inverse. Actually, when this function is of the form $F(x) = Ax - b$ where $A \in \mathbb{R}^{N\times N}$, x and $b \in \mathbb{R}^N$, these methods provide an approximation of A or A^{-1} which can be good candidates to design preconditioners for Krylov subspace methods. Particularly, we take a closer look at variants of two techniques issued from the numerical optimization literature, known as Broyden and EN methods (see, e.g., [23, 48] and [31], respectively). The following presentation is based on the review paper [113] where both families are adapted to solve the possibly nonsymmetric linear system

$$Ax = b. \tag{3.1}$$

They are introduced in their general form in Algorithms 8 and 9, respectively, and involve a matrix $H_k \in \mathbb{R}^{N\times N}$ approximating the inverse of the Jacobian A^{-1} at iteration k.

Algorithm 8 Class of Broyden methods to solve $Ax = b$, using an approximation of A^{-1} [113]

1: Define $x_0 \in \mathbb{R}^N$, $H_0 \in \mathbb{R}^{N\times N}$ and $r_0 = b - Ax_0$
2: **for** $k = 1, \ldots$ **do**
3: $\quad s_{k-1} = H_{k-1} r_{k-1}$
4: $\quad y_{k-1} = A s_{k-1}$
5: $\quad$ Define $\alpha_{k-1} \in \mathbb{R}$
6: $\quad x_k = x_{k-1} + \alpha_{k-1} s_{k-1}$
7: $\quad r_k = r_{k-1} - \alpha_{k-1} y_{k-1}$
8: $\quad$ Define $f_{k-1} \in \mathbb{R}^N$
9: $\quad H_k = H_{k-1} + \frac{(s_{k-1} - H_{k-1} y_{k-1}) f_{k-1}^T}{f_{k-1}^T y_{k-1}}$
10: **end for**

Algorithm 9 EN-like methods to solve $Ax = b$, using an approximation of A^{-1} [113]

Define $x_0 \in \mathbb{R}^N$, $H_0 \in \mathbb{R}^{N\times N}$ and $r_0 = b - Ax_0$
for $k = 1, ...$ **do**
  $s_{k-1} = H_{k-1}r_{k-1}$
  $y_{k-1} = As_{k-1}$
  Define $f_{k-1} \in \mathbb{R}^N$
  $H_k = H_{k-1} + \frac{(s_{k-1}-H_{k-1}y_{k-1})f_{k-1}^T}{f_{k-1}^T y_{k-1}}$
  $\tilde{s}_{k-1} = H_k r_{k-1}$
  $\tilde{y}_{k-1} = A\tilde{s}_{k-1}$
  $x_k = x_{k-1} + \tilde{s}_{k-1}$
  $r_k = r_{k-1} - \tilde{y}_{k-1}$
end for

Different variants of Broyden and EN-like methods are possible, they depend on the definition of the vector f_{k-1} at iteration k, under the assumption that $f_{k-1}^T y_{k-1} \neq 0$. The choice of the coefficient α_{k-1} in Algorithm 8 is not addressed here, since we essentially focus on the expression of H_k as an approximation of A^{-1}. We actually aim at designing a preconditioning technique based on this matrix. We note that the expressions of H_k are identical in both methods, although the descent directions s_{k-1} are not similar. Finally, among the different propositions in [113], we choose $f_{k-1} = y_{k-1} = As_{k-1}$ for two different reasons. First, even if the corresponding Broyden method is known as the "bad" Broyden variant in the literature with lower computational gains in practice, the performance of the related EN-like method in [113, 114] are attractive. In particular, this latter method gives better results than the "good" Broyden variant. Secondly, this choice will allow us to define a class of preconditioners from H_k, without restricting the choice of the vectors s_{k-1} to the descent directions computed during the optimization process. This is the purpose of Propositions 3.1.1, 3.1.2 respectively and Definition 3.1.1 detailed next. Before, let us write H_k differently, introducing the choice $f_{k-1} = As_{k-1}$:

$$H_k = H_{k-1} + \frac{(s_{k-1} - H_{k-1}As_{k-1})s_{k-1}^T A^T}{s_{k-1}^T A^T As_{k-1}},$$
$$H_k = H_{k-1}\Big(I_N - \frac{As_{k-1}s_{k-1}^T A^T}{s_{k-1}^T A^T As_{k-1}}\Big) + \frac{s_{k-1}s_{k-1}^T A^T}{s_{k-1}^T A^T As_{k-1}}. \tag{3.2}$$

As a first step, we want to extend the formula (3.2) to the case where the vectors s_i are assumed to satisfy $s_i^T A^T As_j = 0$ for $i, j \in \{0, ...k-1\}$ and $i \neq j$. We present in the next proposition an expression of this generalization.

Proposition 3.1.1. *Let $S = [s_0, ..., s_{k-1}] \in \mathbb{R}^{N\times k}$ of rank k with column vectors sat-*

isfying $s_i^T A^T A s_j = 0$ *for* $i, j \in \{0, ...k-1\}$ *and* $i \neq j$. *Let us denote* $H_{S,k}$ *the matrix defined in relation* (3.2). $H_{S,k}$ *can be written as*

$$H_{S,k} = H_0(I_N - AS(S^T A^T AS)^{-1} S^T A^T) + S(S^T A^T AS)^{-1} S^T A^T. \tag{3.3}$$

Proof. For the sake of clarity, we denote in this proof $y_i = As_i$ for $i \in \{0, ...k-1\}$. First, we prove by induction the following property

$$(\mathcal{P}_k): \quad H_k = H_0\Big(I_N - \sum_{i=0}^{k-1} \frac{y_i y_i^T}{y_i^T y_i}\Big) + \sum_{i=0}^{k-1} \frac{s_i y_i^T}{y_i^T y_i},$$

where $k \in \mathbb{N}^\star$ and H_k is defined by relation (3.2).

- For $k = 1$, the result directly comes from the definition of H_1.
- Assume that $(\mathcal{P}_{k-1})$ is satisfied for $k \geq 2$.

$$\begin{aligned} H_k &= H_{k-1}(I_N - \frac{y_{k-1} y_{k-1}^T}{y_{k-1}^T y_{k-1}}) + \frac{s_{k-1} y_{k-1}^T}{y_{k-1}^T y_{k-1}}, \\ &= \Big[H_0\Big(I_N - \sum_{i=0}^{k-2} \frac{y_i y_i^T}{y_i^T y_i}\Big) + \sum_{i=0}^{k-2} \frac{s_i y_i^T}{y_i^T y_i}\Big]\Big[I_N - \frac{y_{k-1} y_{k-1}^T}{y_{k-1}^T y_{k-1}}\Big] + \frac{s_{k-1} y_{k-1}^T}{y_{k-1}^T y_{k-1}}. \end{aligned}$$

Since $y_i^T y_{k-1} = 0$ for $i \in \{0, ..., k-2\}$, H_k can be written as

$$\begin{aligned} H_k &= H_0\Big(I_N - \sum_{i=0}^{k-2} \frac{y_i y_i^T}{y_i^T y_i}\Big) - H_0 \frac{y_{k-1} y_{k-1}^T}{y_{k-1}^T y_{k-1}} + \sum_{i=0}^{k-2} \frac{s_i y_i^T}{y_i^T y_i} + \frac{s_{k-1} y_{k-1}^T}{y_{k-1}^T y_{k-1}}, \\ &= H_0\Big(I_N - \sum_{i=0}^{k-1} \frac{y_i y_i^T}{y_i^T y_i}\Big) + \sum_{i=0}^{k-1} \frac{s_i y_i^T}{y_i^T y_i}. \end{aligned}$$

Hence $(\mathcal{P}_k)$ is verified for all $k \in \mathbb{N}^\star$. Moreover, under the assumption $s_i^T A^T A s_j = 0$ for $i \neq j$, we note that

$$(S^T A^T AS)^{-1} = \text{ diag}((y_i^T y_i)^{-1}),$$

which leads to

$$\sum_{i=0}^{k-1} \frac{y_i y_i^T}{y_i^T y_i} = AS(S^T A^T AS)^{-1} S^T A^T \text{ and } \sum_{i=0}^{k-1} \frac{s_i y_i^T}{y_i^T y_i} = S(S^T A^T AS)^{-1} S^T A^T.$$

□

The assumption of A^TA-conjugacy of the vectors s_i in Proposition 3.1.1 is still restrictive. However, Proposition 3.1.2 shows an invariance property of $H_{S,k}$ under a change of basis of $\mathcal{S} = \mathcal{R}(S)$. In other words, one can replace S in (3.3) by a matrix whose column vectors span the same subspace $\mathcal{S}$, and the A^TA-conjugacy assumption can be finally relaxed.

Proposition 3.1.2. *Let $S = [s_0, ..., s_{k-1}] \in \mathbb{R}^{N\times k}$ of rank k and $Z = SX$ where $X \in \mathbb{R}^{k\times k}$ is a nonsingular matrix. Then $H_{Z,k} = H_{S,k}$.*

Proof. Replacing $Z = SX$, we obtain

$$\begin{aligned} AZ(Z^TA^TAZ)^{-1}Z^TA^T &= ASX(X^TS^TA^TASX)^{-1}X^TSTA^T \\ &= ASXX^{-1}(S^TA^TAS)^{-1}X^{-T}X^TS^TA^T \\ &= AS(S^TA^TAS)^{-1}S^TA^T. \end{aligned}$$

Similarly,

$$Z(Z^TA^TAZ)^{-1}Z^TA^T = S(S^TA^TAS)^{-1}S^TA^T.$$

□

Then, since A^TA is symmetric positive definite, we just need to be sure that the k column vectors of S are linearly independent. Setting $H_0 = I_N$, we propose from this generalization of variants of Broyden and EN-like methods a definition of a class of preconditioners adapted to nonsymmetric linear systems.

Definition 3.1.1. *Let $A \in \mathbb{R}^{N\times N}$ be a general nonsingular matrix and $S \in \mathbb{R}^{N\times k}$ of full rank k, with $k \leq N$. The matrix $H \in \mathbb{R}^{N\times N}$ defined by*

$$H = (I_N - AS(S^TA^TAS)^{-1}S^TA^T) + S(S^TA^TAS)^{-1}S^TA^T \tag{3.4}$$

is called the limited memory preconditioner in the nonsymmetric case (LMP_{ns}).

To conclude this section, we analyse the particular case of $k = N$. Obviously, it is of no practical use, since it would necessitate to deal with the inverse of the matrix $S^TA^TAS \in \mathbb{R}^{N\times N}$. Nevertheless, the following proposition shows that H corresponds to A^{-1} in such a case and strengthens the idea to use this class as a preconditioner to improve the convergence rate of the GMRES method.

Proposition 3.1.3. *Let suppose $S \in \mathbb{R}^{N\times N}$ nonsingular. Then $H = A^{-1}$.*

Proof. Under this assumption, S^{-1} exists and

$$H = (I_N - AS(S^TA^TAS)^{-1}S^TA^T) + S(S^TA^TAS)^{-1}S^TA^T,$$
$$H = (I_N - ASS^{-1}A^{-1}A^{-T}S^{-T}S^TA^T) + SS^{-1}A^{-1}A^{-T}S^{-T}S^TA^T,$$
$$H = A^{-1}.$$

□

3.2 Spectral characterization of the preconditioned matrix

The main contribution of this section concerns the characterization of the spectrum of AH, where H is defined by relation (3.4). This spectral analysis begins with the following relation, obtained by a direct calculation:

$$AH = AP_{(A\mathcal{S})^\perp} - I_N - P_{(A\mathcal{S})^\perp}, \tag{3.5}$$

where $P_{(A\mathcal{S})^\perp} = I_N - AS(S^TA^TAS)S^TA^T$ corresponds to the orthogonal projection onto $(A\mathcal{S})^\perp$. From now on, we assume that the columns of $W \in \mathbb{R}^{N\times k}$ and $W_\perp \in \mathbb{R}^{N\times(N-k)}$ form an orthonormal basis for $A\mathcal{S}$ and $(A\mathcal{S})^\perp$, respectively. We characterize $P_{(A\mathcal{S})^\perp}$ and the spectrum of AH in Proposition 3.2.1 and Theorem 3.2.2, respectively.

Proposition 3.2.1. *The operator $P_{(A\mathcal{S})^\perp}$ satisfies the following properties:*

- $P_{(A\mathcal{S})^\perp}W = 0_{N,k}$,
- $P_{(A\mathcal{S})^\perp}W_\perp = W_\perp$.

Proof. The fact that $\mathcal{R}(P_{(A\mathcal{S})^\perp}) = \mathcal{N}(I_N - P_{(A\mathcal{S})^\perp}) = (A\mathcal{S})^\perp$ and $\mathcal{N}(P_{(A\mathcal{S})^\perp}) = \mathcal{R}(I_N - P_{(A\mathcal{S})^\perp}) = A\mathcal{S}$ yields the results. □

Theorem 3.2.2. *Let $A \in \mathbb{R}^{N\times N}$ be a general nonsingular matrix and H defined by relation* (3.4)*. The spectrum of AH is then given by*

$$\Lambda(AH) = \{1\} \cup \Lambda(W_\perp^TAW_\perp).$$

Proof. Since the columns of $[W, W_\perp]$ form an orthonormal basis of $\mathbb{R}^N$, $\Lambda(AH) =$

$\Lambda([W, W_\perp]^T AH[W, W_\perp])$ and

$$[W, W_\perp]^T AH[W, W_\perp] = \begin{pmatrix} W^T AHW & W^T AHW_\perp \\ W_\perp^T AHW & W_\perp^T AHW_\perp \end{pmatrix}.$$

The relation (3.5) and the properties of Proposition 3.2.1 lead to

$$[W, W_\perp]^T AH[W, W_\perp] = \begin{pmatrix} I_k & W^T AW_\perp \\ 0_{N-k,k} & W_\perp^T AW_\perp \end{pmatrix}.$$

□

We deduce from this theorem that 1 is an eigenvalue of AH of multiplicity at least k. The remaining ones are characterized by the spectrum of $W_\perp^T AW_\perp$, revealing the theoretical interest of choosing $\mathcal{S}$ as an A-invariant subspace. The following corollary gives a result in this sense.

Corollary 3.2.3. *Assume that $A \in \mathbb{R}^{N\times N}$ is nonsingular with $\Lambda(A) = \{\lambda_1, ..., \lambda_N\} \in \mathbb{C}^N$. Let us set $S \in \mathbb{R}^{N\times k}$ where $\mathcal{S} = \mathcal{R}(S)$ spans the eigenspace associated to $\{\lambda_1, ..., \lambda_k\}$ which contains some eigenvalues and their conjugate. Then*

$$\Lambda(AH) = \{1, ...1, \lambda_{k+1}, ..., \lambda_N\}.$$

Proof. Under these assumptions, $A\mathcal{S}$ and $(A\mathcal{S})^\perp$ are A-invariant. Finally, the spectral characterization of Theorem 3.2.2 leads to

$$\begin{aligned} \Lambda(AH) &= \{1\} \cup \Lambda(W_\perp^T AW_\perp), \\ \Lambda(AH) &= \{1\} \cup \Lambda(A_{|(A\mathcal{S})^\perp}), \\ \Lambda(AH) &= \{1\} \cup \{\lambda_{k+1}, ..., \lambda_N\}. \end{aligned}$$

□

Remark 10. As mentioned in Section 2.3.3, A has real entries so the conjugate of a complex eigenvalue λ also belongs to $\Lambda(A)$. In Corollary 3.2.3, if $\{\lambda, \bar{\lambda}\} \subset \{\lambda_1, ..., \lambda_k\}$, the associated conjugate eigenvectors s and $\bar{s}$ can be selected. Finally, from Proposition 3.1.2 and the fact that

$$span\{\mathrm{Re}(s), \mathrm{Im}(s)\} = span\{s, \bar{s}\},$$

we can consider that $S \in \mathbb{R}^{N\times k}$.

Unlike the situation of Corollary 3.2.3, only approximations to invariant subspaces can be usually used in practice, as illustrated in Chapter 2. The strategy to select S in the numerical experiments will be based either on Ritz or harmonic Ritz vectors (see Section 3.4). This is also used in some existing methods based for instance on deflation, which are introduced next.

3.3 Theoretical comparison with existing methods

In this section, we aim at positioning and comparing the class of LMP$_{ns}$ with respect to existing methods in the literature. Using a more general approach, we can also include the class of LMP$_{\pm}$, studied in Chapter 2 in the symmetric indefinite case. They are actually interrelated in the sense that they involve similar operators, summarized below.

1. The LMP$_{\pm}$ for symmetric indefinite linear systems:

$$H_{\pm} = (I_N - S(S^TAS)^{-1}S^TA)(I_N - AS(S^TAS)^{-1}S^T) + S(S^TAS)^{-1}S^T, \quad (3.6)$$

   where $A \in \mathbb{R}^{N\times N}$ is symmetric indefinite and $S \in \mathbb{R}^{N\times k}$ such that S^TAS is nonsingular. $H_{\pm}$ involves the action of:

   - the oblique projector $P_{\mathcal{S}^{\perp},A\mathcal{S}} = I_N - AS(S^TAS)^{-1}S^T$ and $P^T_{\mathcal{S}^{\perp},A\mathcal{S}}$,
   - the "shift" term $S(S^TAS)^{-1}S^T$.

2. The LMP$_{ns}$ for nonsingular linear systems:

$$H_{ns} = (I_N - AS(S^TA^TAS)^{-1}S^TA^T) + S(S^TA^TAS)^{-1}S^TA^T, \quad (3.7)$$

   where $A \in \mathbb{R}^{N\times N}$ is nonsingular and $S \in \mathbb{R}^{N\times k}$ is of full rank k. H_{ns} involves the action of:

   - the orthogonal projector $P_{(A\mathcal{S})^{\perp}} = I_N - AS(S^TA^TAS)^{-1}S^TA^T$,
   - the "shift" term $S(S^TA^TAS)^{-1}S^TA^T$.

These preconditioning techniques are based on a generalization of methods proposed in the numerical optimization literature as described in Sections 2.1.1 and 3.1, but the operators given above also appear in different references in the literature. Two of them are introduced next and their action on the convergence of GMRES is investigated.

3.3.1 Deflation

The projectors $P_{\mathcal{S}^\perp,A\mathcal{S}}$ and $P_{(A\mathcal{S})^\perp}$ are particularly related to deflation in the literature. We refer the reader to [38, 39, 51, 99] for a comprehensive theoretical overview on these methods and to references therein for a summary of applications, where the relevance of these methods has been shown. Several authors have proposed deflated Krylov subspace methods with applications in different fields of numerical analysis. We can cite [29] or [92] for symmetric positive definite problems using the Conjugate Gradient method, or [33] with GMRES for nonsingular systems. This section is based on the general approach used in this latter review paper which allows to take into account both $\text{LMP}_\pm$ and LMP_{ns} within the context of GMRES. For the sake of clarity, we consider in Section 3.3 that the initial guess x_0 is always zero.

For S and $Y \in \mathbb{R}^{N\times k}$ such that Y^TAS is nonsingular, let us denote

$$\begin{aligned} P_D &= I_N - AS(Y^TAS)^{-1}Y^T, \\ Q_D &= I_N - S(Y^TAS)^{-1}Y^TA, \\ R_D &= S(Y^TAS)^{-1}Y^T. \end{aligned} \tag{3.8}$$

We easily note the relation between these operators and both LMPs:

1. In case of $Y = S$ and A symmetric indefinite, the $\text{LMP}_\pm$ defined in relation (2.6) satisfies
$$H_\pm = Q_DP_D + R_D.$$
2. In case of $Y = AS$, the LMP_{ns} defined in relation (3.4) satisfies
$$H_{ns} = P_D + R_D.$$

The main steps of the deflation methods are now introduced, but we prior expose some useful properties of the operators defined in (3.8), obtained from direct calculations.

Proposition 3.3.1. *Let P_D, Q_D and R_D defined in* (3.8).

- $P_D^2 = P_D,$ (3.9)
- $Q_D^2 = Q_D,$ (3.10)
- $AQ_D = P_DA,$ (3.11)

- $R_D A = I_N - Q_D,$ (3.12)
- $AR_D = I_N - P_D,$ (3.13)
- $R_D P_D = 0_{N\times N}.$ (3.14)

The deflation is based on the decomposition of the solution x of (3.1) into

$$\begin{aligned} x &= (I_N - Q_D)x + Q_D x, \\ &= R_D A x + Q_D x, \\ &= R_D b + Q_D x. \end{aligned}$$

The first term $R_D b$ is then easily obtained and it remains to compute $Q_D x$. The original linear system (3.1) can be multiplied by the projector P_D and, using the equalities (3.9) and (3.11) in Proposition 3.3.1, we obtain:

$$\begin{aligned} P_D A x = P_D b &\Leftrightarrow P_D^2 A x = P_D b \\ &\Leftrightarrow P_D A Q_D x = P_D b. \end{aligned}$$

Finally, the key is to solve the deflated system

$$\begin{cases} P_D A\tilde{x} = P_D b \text{ using GMRES with } \tilde{x}_0 = 0, \\ x = Q_D\tilde{x} + R_D b. \end{cases} . \tag{3.15}$$

The matrix P_D being a projector, $P_D A$ is singular but equation (3.15) is consistent in the sense that $P_D b \in \mathcal{R}(P_D A)$. Hence, it can be solved with a Krylov subspace method such as GMRES (see, e.g., [40]). In other words, the solution of $P_D A\tilde{x} = P_D b$ is not necessarily a solution of $Ax = b$ but the addition of the correction term $(I_N - Q_D)x = R_D b$ to $Q_D x = Q_D\tilde{x}$ yields the unique solution of the original system.

3.3.2 Abstract balancing preconditioner

In [33], the authors compare the deflated GMRES method with an abstract balancing preconditioner for nonsymmetric linear systems. This latter is an extension of the abstract form of the coarse grid correction preconditioner used in domain decomposition methods, initially proposed in the symmetric positive definite case [107, 67, 68, 104].

Keeping the assumptions and notation of the previous section, this preconditioner is simply defined in [33] as

$$P_B = Q_D P_D + R_D. \tag{3.16}$$

We immediately note the equivalence of P_B with $H_\pm$ when $Y = S$ and A is symmetric indefinite. Concerning H_{ns} when $Y = AS$, the expression is slightly different, since they both involve the action of the projector P_D and the "shift" term R_D.

3.3.3 Spectral comparison

Considering the relations exhibited from the definition of $\text{LMP}_\pm$ and of LMP_{ns} with the deflation operator P_D and the abstract balancing preconditioner P_B, it is interesting to compare the convergence behaviour of GMRES using these different convergence acceleration techniques. As a first step, we analyse the spectra of the associated deflated or preconditioned operators, for A symmetric indefinite with $Y = S$, and A nonsingular with $Y = AS$.

3.3.3.1 Symmetric indefinite case with $Y = S$

In this case, $H_\pm = P_B$ and we note that an analysis has already been proposed in [33]. However, if we include the spectral characterization shown in Theorem 2.2.2, we obtain the following theorem.

Theorem 3.3.2. *Let $A \in \mathbb{R}^{N\times N}$ be a symmetric indefinite matrix and $Y = S \in \mathbb{R}^{N\times k}$ such that $Y^T AS$ is nonsingular. Let us consider $H_\pm$, P_D and P_B given by relations* (3.6), (3.8) *and* (3.16)*, respectively. Moreover, assume that the columns of $Z \in \mathbb{R}^{N\times k}$ form an orthonormal basis for $\mathcal{S} = \mathcal{R}(S)$ and that the columns of $Z_\perp \in \mathbb{R}^{N\times(N-k)}$ form an orthonormal basis for $\mathcal{S}^\perp$. The spectra of the operators $AH_\pm$, $P_D A$ and AP_B are then given by*

$$\Lambda(AH_\pm) = \Lambda(AP_B) = \{1\} \cup \Lambda((Z_\perp^T A^{-1} Z_\perp)^{-1}),$$
$$\Lambda(P_D A) = \{0\} \cup \Lambda((Z_\perp^T A^{-1} Z_\perp)^{-1}).$$

Proof. Theorem 2.2.2 and the equality $H_\pm = P_B$ lead to the spectral characterization of the associated preconditioned matrices. Theorem 2.8 in [33] completes the proof. □

Thus the action of $H_\pm = P_B$ and P_D on A generates almost identical spectra. They differ from the value of the k clustered eigenvalues, equal to 1 with the preconditioning technique and 0 with the deflation.

3.3.3.2 Nonsymmetric case with $Y = AS$

Here, the LMP_{ns} is no more equal to the associated abstract balancing preconditioner. Nevertheless, Theorem 3.3.3 shows that the preconditioned spectra are similar as in the symmetric indefinite case.

Theorem 3.3.3. *Let $A \in \mathbb{R}^{N\times N}$ be a nonsingular matrix, $S \in \mathbb{R}^{N\times k}$ of full rank k and $Y = AS$. Let us consider H_{ns}, P_D and P_B given by relations* (3.7), (3.8) *and* (3.16)*, respectively. Moreover, assume that the columns of $W \in \mathbb{R}^{N\times k}$ form an orthonormal basis for $A\mathcal{S} = \mathcal{R}(AS)$ and that the columns of $W_\perp \in \mathbb{R}^{N\times(N-k)}$ form an orthonormal basis for $A\mathcal{S}^\perp$. The spectra of the operators AH_{ns}, P_DA and AP_B are then given by*

$$\begin{aligned}\Lambda(AH_{ns}) = \Lambda(AP_B) &= \{1\} \cup \Lambda(W_\perp^T AW_\perp),\\ \Lambda(P_DA) &= \{0\} \cup \Lambda(W_\perp^T AW_\perp).\end{aligned}$$

Proof. The spectrum AH_{ns} has already been exhibited in Theorem 3.2.2. Using Theorem 2.8 in [33], we just need to characterize $\Lambda(P_DA)$ to finalize the proof. Since the columns of $[W, W_\perp]$ form an orthonormal basis of $\mathbb{R}^N$, $\Lambda(P_DA) = \Lambda([W, W_\perp]^T P_DA[W, W_\perp])$ and

$$[W, W_\perp]^T P_DA[W, W_\perp] = \begin{pmatrix} W^T P_DAW & W^T P_DAW_\perp \\ W_\perp^T P_DAW & W_\perp^T P_DAW_\perp \end{pmatrix}.$$

Here, $Y = AS$ and $P_D = P_{(A\mathcal{S})^\perp}$ is the orthogonal projection onto $(A\mathcal{S})^\perp$ and the properties of Proposition 3.2.1 remain valid. So

$$[W, W_\perp]^T P_DA[W, W_\perp] = \begin{pmatrix} 0_{k,k} & 0_{k,N-k} \\ W_\perp^T AW & W_\perp^T AW_\perp \end{pmatrix}.$$

□

3.3.4 Comparison of GMRES iterates

The goal is now to compare the successive iterates obtained within GMRES either on the deflated system or when using the preconditioners P_B, $H_\pm$ or H_{ns} on the original linear system. We first focus on the difference between the use of the balancing preconditioner (notably equivalent to the $\text{LMP}_\pm$ under appropriate assumptions) and the deflation.

In [33], the authors show that the 2-norms of the successive actual residuals of GMRES combined with deflation are never larger than those obtained using P_B as a left preconditioner, under different conditions related to the choice of the initial vectors

or Z and Y (see Theorems 3.4 and 3.6). We contribute in this section in analysing what happens when P_B is applied as a right preconditioner. In this sense, we need intermediate results which permit subsequently to draw comparisons between the Krylov subspaces involved in the GMRES process.

Lemma 3.3.4. *Let $A \in \mathbb{R}^{N\times N}$ be a nonsingular matrix and S, $Y \in \mathbb{R}^{N\times k}$ such that $Y^T AZ$ is nonsingular. If we consider the matrices P_D and P_B defined by relations* (3.8) *and* (3.16)*, we obtain*

1. *$P_B P_D = Q_D P_D$,*
2. *For $m \in \mathbb{N}$, $(AP_B)^m P_D = (P_D A)^m P_D$.*

Proof. 1. By relation (3.16), $P_B = Q_D P_D + R_D$. The properties (3.9) and (3.14) in Proposition 3.3.1 ensure the first result.

2. The proof can be done by induction.

   - For $m = 0$, the result is trivial.
   - For $m = 1$, the first result of this theorem and the equality (3.11) lead to

$$\begin{aligned} AP_B P_D &= AQ_D P_D \\ &= P_D A P_D. \end{aligned}$$

   - Suppose that $(AP_B)^{m-1} P_D = (P_D A)^{m-1} P_D$ for $m - 1 \geq 1$. Then, using the same arguments as in the case of $m = 1$, we obtain

$$\begin{aligned} (AP_B)^m P_D &= AP_B (AP_B)^{m-1} P_D \\ &= AP_B (P_D A)^{m-1} P_D \\ &= AP_B P_D A (P_D A)^{m-2} P_D \\ &= P_D A P_D A (P_D A)^{m-2} P_D \\ &= (P_D A)^m P_D. \end{aligned}$$

□

From now on, we denote by $x_{D,k}$ and $x_{B,k}$ the successive iterates at iteration k obtained within GMRES, using the deflation and the balancing preconditioner, respectively. From relation (3.15) and Section 1.2, we know that the iterates of the deflated

GMRES belong to the affine subspaces

$$x_{D,k} \in R_D b + Q_D \; span\{P_D b, (P_D A)P_D b, ..., (P_D A)^{k-1} P_D b\}.$$

On the other hand, using GMRES with P_B as a right preconditioner leads to

$$x_{B,k} \in x_{B,0} + P_B \; span\{b, (AP_B)b, ..., (AP_B)^{k-1} b\}.$$

In order to obtain similar iterates, the choice $x_{B,0} = R_D b$ seems to be natural. In this case, using the relation (3.13), we can transform the original linear system into

$$\begin{aligned} Ax = b &\Longleftrightarrow A(x - R_D b) = b - AR_D b, \\ &\Longleftrightarrow A(x - R_D b) = P_D b, \\ &\Longleftrightarrow \left\{ \begin{array}{l} AP_B \tilde{x} = P_D b \\ x = P_B \tilde{x} + R_D b. \end{array} \right. \end{aligned} \tag{3.17}$$

Adapting the notation to the case of the balancing preconditioner, we assume that $\tilde{x}_{B,0} = 0$ in relation (3.17). So, the related GMRES iterates belong to

$$x_{B,k} \in R_D b + P_B \; span\{P_D b, (AP_B)P_D b, ..., (AP_B)^{k-1} P_D b\}.$$

Both results in Lemma 3.3.4 finally lead to the same affine subspace involved when using GMRES on the deflated system:

$$x_{B,k} \in R_D b + Q_D \; span\{P_D b, (P_D A)P_D b, ..., (P_D A)^{k-1} P_D b\}.$$

In this context, the following theorem summarizes the equivalence between the deflation and balancing preconditioners.

Theorem 3.3.5. *Let $A \in \mathbb{R}^{N \times N}$ be a nonsingular matrix and S, $Y \in \mathbb{R}^{N \times k}$ such that $Y^T AS$ is also nonsingular. GMRES combined with deflation in relation* (3.15) *for $\tilde{x}_{D,0} = 0$ and with the balancing preconditioner in relation* (3.17) *for $\tilde{x}_{B,0} = 0$ produces the same iterates in exact arithmetics, i.e. $x_{D,k} = x_{B,k}$ for all k.*

Proof. Under these assumptions and according to the previous analysis, the GMRES iterates related to the linear systems (3.15) and (3.17) can respectively be written as

$$\begin{aligned} x_{D,k} &= R_D b + Q_D \tilde{x}_{D,k}, \\ x_{B,k} &= R_D b + P_B \tilde{x}_{B,k}, \end{aligned}$$

where $\tilde{x}_{D,k}$, $\tilde{x}_{B,k} \in \mathcal{K}(P_D A, P_D b) = span\{P_D b, (P_D A)P_D b, ..., (P_D A)^{k-1} P_D b\}$. According to Algorithm 2, the first step in GMRES aims at computing an orthonormal basis of this Krylov subspace using the Arnoldi method. Since $\tilde{x}_{B,k} = \tilde{x}_{B,k} = 0$, the associated initial residuals are both equal to $P_D b$. The Arnoldi process applied to (3.15) and (3.17) generates the same Arnoldi matrices V_k and Hessenberg matrices $H_{k+1,k}$ in exact arithmetics. Therefore it follows that the respective least-squares problems to be solved are identical, leading to

$$y^\star = \underset{y \in \mathbb{R}^k}{\operatorname{argmin}} \| \frac{P_D b}{\|P_D b\|_2} e_1 - H_{k+1,k} y \|_2.$$

Finally, the approximated solutions of the original system (3.1) are obtained by

$$\begin{aligned} x_{D,k} &= R_D b + Q_D V_k y^\star, \\ x_{B,k} &= R_D b + P_B V_k y^\star. \end{aligned}$$

Here, the column vectors of V_k form an orthonormal basis of $\mathcal{K}(P_D A, P_D b)$. Since P_D is a projector, each vector x in this subspace satisfies $P_D x = x$. In particular, $V_k = P_D V_k$ and the first result in Lemma 3.3.4 gives

$$\begin{aligned} x_{B,k} &= R_D b + P_B V_k y^\star \\ &= R_D b + P_B P_D V_k y^\star \\ &= R_D b + Q_D P_D V_k y^\star \\ &= R_D b + Q_D V_k y^\star \\ &= x_{D,k}. \end{aligned}$$

□

Theorem 3.3.5 states an equivalence between the deflation and the balancing preconditioner used as a right preconditioner for GMRES. The associated methods are detailed in Algorithms 10 and 11, respectively. In both algorithms, we use a slightly more general framework, introducing the restart variant and possible nonzero initial approximations of the solution. However, the equivalence property remains obviously valid.

The relation with the $\text{LMP}_\pm$ is immediate since $H_\pm = P_B$ when $Y = S$ and A is symmetric. Concerning the LMP_{ns}, there is no equivalence statement to the best of our knowledge. However, Theorem 3.3.3 reveals the spectral closeness of the respective

deflated or preconditioned operators. Since the spectrum alone cannot describe the convergence rate of GMRES for general nonsingular systems, it is interesting to compare numerically the action of these techniques. This is performed in Section 3.5. Before, we discuss some implementation issues, focusing on the computational cost and on the choice of the matrix S.

Algorithm 10 Deflated GMRES(m) to solve $Ax_D = b$

Choose a convergence threshold ϵ
Choose $x_{D,0} \in \mathbb{R}^N$, S and $Y \in \mathbb{R}^{N\times k}$
Define $P_D \in \mathbb{R}^{N\times N}$ as $P_D = I_N - AS(Y^TAS)^{-1}Y^T$
Define $Q_D \in \mathbb{R}^{N\times N}$ as $Q_D = I_N - S(Y^TAS)^{-1}Y^TA$
Define $R_D \in \mathbb{R}^{N\times N}$ as $R_D = S(Y^TAS)^{-1}Y^T$
Define $x_{D,0}^{(1)} = x_{D,0}$ and $r_{D,0}^{(1)} = b - Ax_{D,0}^{(1)}$
for $l = 1, \ldots$ **do**
  # Solution phase of the deflated linear system $P_DA\tilde{x}_D = P_Dr_{D,0}^{(l)}$ with $\tilde{x}_{D,0} = 0$
  Set $\beta = \|P_Dr_{D,0}^{(l)}\|_2$ and $v_1 = P_Dr_{D,0}^{(l)}/\beta$
  Apply m steps of Arnoldi to obtain $V_{m+1} = [V_m, v_{m+1}] \in \mathbb{R}^{N\times(m+1)}$ and $H_{m+1,m} \in \mathbb{R}^{(m+1)\times m}$ such that:

$$P_DAV_m = V_{m+1}H_{m+1,m}$$

  Solve the least-squares minimization problem

$$y^\star = \underset{y\in\mathbb{R}^m}{\operatorname{argmin}} \|\beta e_1 - H_{m+1,m}y\|_2$$

  Define $\tilde{x}_D = V_my^\star \in \mathbb{R}^N$ and $\tilde{r}_D = V_{m+1}(\beta e_1 - H_{m+1,m}y^\star) \in \mathbb{R}^N$
  if $\|\beta e_1 - H_{m+1,m}y^\star\|_2 \leq \epsilon \times \beta$ **then**
    stop
  end if
  # Correction phase
  $x_{D,0}^{(l+1)} = x_{D,0}^{(l)} + Q_D\tilde{x}_D + R_Dr_{D,0}^{(l)}$
  $r_{D,0}^{(l+1)} = \tilde{r}_D$
end for

Algorithm 11 GMRES(m) with a right balancing preconditioner to solve $Ax_B = b$ as described in (3.17)

Choose a convergence threshold ϵ
Chose $x_{B,0} \in \mathbb{R}^N$, S and $Y \in \mathbb{R}^{N\times k}$
Define $P_D \in \mathbb{R}^{N\times N}$ as $P_D = I_N - AS(Y^TAS)^{-1}Y^T$
Define $P_D \in \mathbb{R}^{N\times N}$ as $Q_D = I_N - S(Y^TAS)^{-1}Y^TA$
Define $R_D \in \mathbb{R}^{N\times N}$ as $R_D = S(Y^TAS)^{-1}Y^T$
Define $x_{B,0}^{(1)} = x_{B,0}$ and $r_0^{(1)} = b - Ax_{B,0}^{(1)}$
\# Definition of the balancing preconditioner
\# (equivalent to the LMP$_\pm$ for $Y = S$ and A symmetric)
$P_B = Q_DP_D + R_D$
for $l = 1, ...$ **do**
    \# Solution phase of the linear system $AP_B\tilde{x}_B = P_Dr_{B,0}^{(l)}$ using P_B as a right preconditioner and $\tilde{x}_{B,0} = 0$.
    Set $\beta = \|P_Dr_{B,0}^{(l)}\|_2$ and $v_1 = P_Dr_{B,0}^{(l)}/\beta$
    Apply m steps of Arnoldi to obtain $V_{m+1} = [V_m, v_{m+1}] \in \mathbb{R}^{N\times(m+1)}$ and $H_{m+1,m} \in \mathbb{R}^{(m+1)\times m}$ such that:

$$AP_BV_m = V_{m+1}H_{m+1,m}$$

    Solve the least-squares minimization problem

$$y^\star = \underset{y\in\mathbb{R}^m}{\operatorname{argmin}} \|\beta e_1 - H_{m+1,m}y\|_2$$

    Define $\tilde{x}_B = P_BV_my^\star \in \mathbb{R}^N$ and $\tilde{r}_B = V_{m+1}(\beta e_1 - H_{m+1,m}y^\star) \in \mathbb{R}^N$
    if $\|\beta e_1 - H_{m+1,m}y^\star\|_2 \leq \epsilon \times \beta$ **then**
        stop
    end if
    \# Correction phase
    $x_{B,0}^{(l+1)} = x_{B,0}^{(l)} + \tilde{x}_B + R_Dr_{B,0}^{(l)}$
    $r_{B,0}^{(l+1)} = \tilde{r}_B$
end for

3.4 Implementation considerations

3.4.1 Choice of the matrix S

As mentioned in the previous sections of this chapter, the definition of H_{ns} in (3.4) (or P_D in (3.8) and P_B in (3.16)) is related to the choice of the matrix S. Here, we consider using GMRES(30) for the solution of a sequence of the form

$$A_i x_i = b_i,$$

where the nonsymmetric matrices are supposed to slowly change. Similarly to Chapter 2, the strategy to select column vectors of S is based either on Ritz or harmonic Ritz vectors, recovered during the solution of the first linear system $A_1 x_1 = b_1$. In practice, these vectors are computed from the Hessenberg matrix (see Section 1.2.3), which is obtained during the last complete cycle of GMRES(30) (or during the first cycle if only one is required). In the nonsymmetric case, this matrix is also nonsymmetric, and both Ritz and harmonic Ritz vectors can be complex-valued. Nevertheless, since H_{ns} (as well as P_B and P_D) is invariant under a change of basis of $\mathcal{S}$, we use the trick presented in Section 2.3.3 to obtain a real-valued matrix S. In other words, if we want to select a vector associated to a complex Ritz or harmonic Ritz value, we impose to select both real and imaginary parts of this vector as columns of S. Hence, for a given integer k, we may allow the method to define $S \in \mathbb{R}^{N\times(k+1)}$.

Remark 11. From now on, we use the term Ritz or harmonic Ritz vectors, even if the vector corresponds to the real or the imaginary part of a given vector.

3.4.2 Computational cost and memory requirements of the LMP$_{ns}$

We consider now that $S \in \mathbb{R}^{N\times k}$ is defined. We propose a possible implementation of the LMP$_{ns}$ matrix H_{ns} given in (3.4). Here, in order to "eliminate" the term $(S^T A^T A S)^{-1}$, we use the invariance property of H_{ns} under a change of basis of $\mathcal{S}$ and the fact that $A^T A$ is symmetric positive definite. Actually, an orthonormalization process based on Gram-Schmidt can be performed, defining a matrix $Z \in \mathbb{R}^{N\times k}$ such that $\mathcal{R}(Z) = \mathcal{R}(S)$ and $Z^T A^T A Z = I_N$. Denoting $X = AZ$, H_{ns} can then be written as

$$H_{ns} = I_N - XX^T + ZX^T.$$

We further note that this expression can be written as

$$H_{ns} = I_N + YX^T, \tag{3.18}$$

where $Y = Z - X$. Algorithm 12 details this process, where $Z_i = [z_1, ..., z_i]$ and $X_i = [x_1, ..., x_i]$. This algorithm requires, in terms of computational cost, k matrix-vector products by A and $4N + \sum_{i=1}^{k-1}(6iN + 6N) + kN = (3k^2 + 4k - 2)N$ additional floating point operations, as well as $2k$ vectors of length N to store. Each application of H_{ns} on a vector can be directly performed in $(4k+1)N$ floating point operations, using relation (3.18).

Remark 12. To the best of our knowledge, the use of Ritz or harmonic Ritz vectors does not allow to decrease the computational cost and memory requirements proposed here.

Algorithm 12 Compute Y, $X \in \mathbb{R}^{N\times k}$ such that $H_{ns} = I_N + YX^T$

$z_1 = s_1$
$x_1 = Az_1$ (one product by A)
$\sigma = \|x_1\|_2$ ($2N$ flops)
$x_1 = x_1/\sigma$ (N flops)
$z_1 = z_1/\sigma$ (N flops)
for $i = 1$ to $k-1$ **do**
  $z_{i+1} = s_{i+1}$
  $x_{i+1} = Az_{i+1}$ (one product by A)
  $f = X_i^T x_{i+1}$ ($2iN$ flops)
  $z_i = z_{i+1} - Z_i f$ ($2iN + N$ flops)
  $x_i = x_{i+1} - X_i f$ ($2iN + N$ flops)
  $\sigma = \|x_{i+1}\|_2$ ($2N$ flops)
  $x_{i+1} = x_{i+1}/\sigma$ (N flops)
  $z_{i+1} = z_{i+1}/\sigma$ (N flops)
end for
$Y = X - Z$ (kN flops)

3.4.3 Computational cost and memory requirements of the deflation

In the numerical experiments illustrated in Section 3.5, we will particularly compare the effect of the LMP_{ns} with the deflation method. To apply the deflation operator P_D, as well as Q_D and R_D, we can use the same matrices X and Z computed in Algorithm 12.

Hence, we obtain

$$\begin{cases} P_D = I_N - XX^T, \\ Q_D = I_N - ZX^T A, \\ R_D = ZX^T. \end{cases}$$

Here, we do not need to compute the matrix Y in Algorithm 12 and the orthonormalization process requires k matrix-vector products by A and $(3k^2 + 3k - 2)N$ additional floating point operations. In terms of memory requirements, the deflation method also needs to store $2k$ vectors of length N. Each application of P_D on a vector can also be performed in $(4k + 1)N$ floating point operations. We also note that, at the end of each cycle of GMRES(30), we need to compute the approximation of the solution of the linear system using Q_D and R_D (see Algorithm 10). This step costs 1 matrix-vector product by A and $(8k + 1)N$ floating point operations.

Remark 13. Concerning the abstract balancing preconditioner P_B defined by relation (3.16), we will just analyse its action on a small-scale case (see Section 3.5.2), where only the convergence histories of GMRES(30) are compared. More precisely, we illustrate on this problem the equivalence stated in Theorem 3.3.5. Hence, we do not detail the computational cost and the memory requirements in this case.

3.5 Applications to solid mechanics

The purpose of this section is to illustrate the efficiency of the class of LMP$_{ns}$ on systems of saddle point structure in solid mechanics introduced in Section 1.1. The method has been implemented in Code_Aster via the PETSc library (version 3.4.5), and a description of the routines is available in Appendix A.

3.5.1 Sequence of preconditioned saddle point systems

We consider, in the next numerical experiments, mechanical problems leading to sequences of symmetric indefinite linear systems, already defined before in this manuscript. The sequences are of the form

$$\mathcal{K}_i \; y_i = c_i \Longleftrightarrow \begin{pmatrix} G_i & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u_i \\ \lambda_i \end{pmatrix} = \begin{pmatrix} f_i \\ g_i \end{pmatrix}, \quad i = 1, \cdots, I. \tag{3.19}$$

Contrary to the study led in Chapter 2, the class of LMP$_{ns}$ is adapted to solve nonsymmetric problems and we can solve (3.19) using any first level preconditioner. Two such techniques are used next:

- **A block upper triangular preconditioner**
  In Sections 3.5.2 and 3.5.3, we consider the specific block upper triangular preconditioner based on the augmented Lagrangian method and defined by relation (1.23):

$$\mathcal{M}_t = \begin{pmatrix} G_1 + \gamma B^T B & 2B^T \\ 0 & -\frac{1}{\gamma} I_m \end{pmatrix}, \quad \gamma > 0. \tag{3.20}$$

  $\mathcal{M}_t$ is computed from the first matrix $\mathcal{K}_1$ in (3.19) and is fixed all along the sequence. As justified in Section 2.4.1, we consider an approximation of $\mathcal{M}_t$ based on the incomplete Cholesky factorization of $G_1 + \gamma B^T B$ written as $G_1 + \gamma B^T B \approx LL^T$. Finally, the associated matrix is used as a left preconditioner [1] and we obtain the preconditioned linear systems $\mathcal{A}_i x_i = b_i$, denoted as

$$\mathcal{A}_i\, x_i = b_i \Longleftrightarrow \begin{pmatrix} LL^T & 2B^T \\ 0 & -\frac{1}{\gamma} I_m \end{pmatrix}^{-1} \begin{pmatrix} G_i & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u_i \\ \lambda_i \end{pmatrix} = \begin{pmatrix} LL^T & 2B^T \\ 0 & -\frac{1}{\gamma} I_m \end{pmatrix}^{-1} \begin{pmatrix} f_i \\ g_i \end{pmatrix} \quad i = 1, \cdots, I. \tag{3.21}$$

  In practice, the preconditioner is applied using the relation

$$\begin{pmatrix} LL^T & 2B^T \\ 0 & -\frac{1}{\gamma} I_m \end{pmatrix}^{-1} = \begin{pmatrix} L^{-T} L^{-1} & 0 \\ 0 & -\gamma I_m \end{pmatrix} \begin{pmatrix} I_n & 2\gamma B^T \\ 0 & I_m \end{pmatrix}.$$

  As a remark, the computational cost related to the application of this preconditioner is approximately the same that for the block diagonal variant used in Section 2.4 (the matrix B being very sparse). In practice, we will see later that the block triangular variant is more efficient.

- **Factorization in single precision arithmetics**
  In Sections 3.5.4 and 3.5.5, we consider another first level preconditioner. In these experiments, we use the default preconditioning technique in Code_Aster, named *LDLT_SP* and already introduced in Section 1.3.2.1. This method consists in computing a complete factorization of the saddle point matrix $\mathcal{K}_1$ in single precision arithmetics using the MUMPS library. We use the computed matrix, denoted $\mathcal{M}_{sp}$, as a left preconditioner [2] which leads to the sequence of preconditioned linear systems

$$\mathcal{A}_i\ x_i = b_i \Longleftrightarrow \mathcal{M}_{sp}^{-1} \mathcal{K}_i x_i = \mathcal{M}_{sp}^{-1} b_i \quad i = 1, \cdots, I. \tag{3.22}$$

1. To be in agreement with all preconditioning strategies in Code_Aster.
2. See 1.

$\mathcal{M}_{sp}^{-1}$ is applied by successive back and forward substitutions and, unless otherwise stated, $\mathcal{M}_{sp}$ is fixed all along the sequence (3.22).

GMRES(30) combined with LMP$_{ns}$ as a right preconditioner will be used to solve the sequence of nonsymmetric linear systems (3.21) or (3.22). To select $S \in \mathbb{R}^{N\times k}$, we extract k (harmonic) Ritz vectors corresponding to the smallest in modulus (harmonic) Ritz values, when solving the first linear system of the sequence. The limited memory preconditioner H_{ns} is then defined for all the remaining linear systems as

$$H_{ns} = I_{n+m} - \mathcal{A}_1 S(S^T \mathcal{A}_1^T \mathcal{A}_1 S)^{-1} S^T \mathcal{A}_1^T + S(S^T \mathcal{A}_1^T \mathcal{A}_1 S)^{-1} S^T \mathcal{A}_1^T. \tag{3.23}$$

To summarize, we use GMRES(30) to solve:

$$\mathcal{A}_1 x_1 = b_1 \quad \text{and} \quad \left\{ \begin{array}{ll} \mathcal{A}_i H_{ns} \tilde{x}_i = b_i & \\ x_i = H_{ns} \tilde{x}_i & \end{array} \right. \quad i = 2, \cdots, I.$$

In addition, we will compare this preconditioning technique with the deflation method involving the same matrix S. Contrary to the LMP$_{ns}$ which is a preconditioner, the operators P_D, Q_D and R_D given by relation (3.8) require to be defined from the current matrix (see the presentation in Section 3.3.1). In other words, although S is fixed all along the sequence, we need to define new operators for each linear system if the matrices $\mathcal{A}_i$ vary. Denoting these matrices P_D^i, Q_D^i and R_D^i, we obtain

$$\left\{ \begin{array}{l} P_D^i = I_{n+m} - \mathcal{A}_i S(S^T \mathcal{A}_i^T \mathcal{A}_i S)^{-1} S^T \mathcal{A}_i^T, \\ Q_D^i = I_{n+m} - S(S^T \mathcal{A}_i^T \mathcal{A}_i S)^{-1} S^T \mathcal{A}_i^T \mathcal{A}_i, \\ R_D^i = S(S^T \mathcal{A}_i^T \mathcal{A}_i S)^{-1} S^T \mathcal{A}_i^T. \end{array} \right. \tag{3.24}$$

Finally, with changing matrices $\mathcal{A}_i$, the deflation method requires to perform the orthogonalization process described in Section 3.4 before solving the current deflated linear system. The sequence to solve using GMRES(30) is then:

$$\mathcal{A}_1 x_1 = b_1 \quad \text{and} \quad \left\{ \begin{array}{ll} P_D^i \mathcal{A}_i \tilde{x}_i = P_D^i b_i & \\ x_i = Q_D^i \tilde{x}_i + R_D^i b_i & \end{array} \right. \quad \text{for } i = 2, \cdots, I.$$

We choose a fixed stopping criterion for each linear system, which is defined as

$$\frac{||b_i - \mathcal{A}_i x_i^k||_2}{||b_i||_2} \leq 10^{-8}, \tag{3.25}$$

and a zero initial guess x_i^0 is always chosen. We further note that the numerical results have been obtained on the Aster5 cluster, described in Section 2.4.

3.5.2 Mechanical bearing (linear case)

We first deal with the linear problem related to the displacement of a mechanical bearing, already introduced in Section 2.4.2. The solution method is different, notably due to the choice of the first level preconditioner: here, we use an approximation of the block upper triangular (3.20) which leads to solve one nonsymmetric system $\mathcal{A}x = b$ in (3.21). However, the parameter γ and the incomplete Cholesky factorization of the matrix $G_1 + \gamma B^T B$ are similar to Section 2.4.2.

Figure 3.1 shows the spectrum of the matrix $\mathcal{A}$ (computed within the SLEPc library). This illustration differs from Figure 2.6, since the information about the respective first level preconditioners are taken into account. Here, we note that the real part of the spectrum is strictly positive and that a cluster around 1 appears. This eigendistribution can be related to the spectral characterization of Theorem 1.3.2, where the exact augmented matrix $G_1 + \gamma B^T B$ is used.

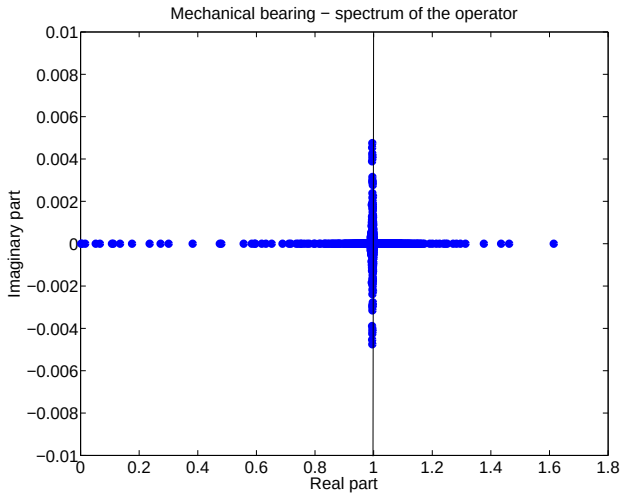

Figure 3.1: Mechanical bearing: spectrum of $\mathcal{A}$.

First, we investigate the behaviour of the class of LMP$_{ns}$ combined with the block triangular augmented Lagrangian preconditioner. Each panel in Figure 3.2 gathers the convergence history of GMRES(30) without LMP$_{ns}$, or with LMP$_{ns}$ based on Ritz,

harmonic Ritz and exact eigenvectors for a same number k of columns in S ($k = 5$, 20 or 30). These selected vectors are all associated to the smallest in modulus Ritz, harmonic Ritz or eigenvalues. The first remark concerns the effect of the first level preconditioner alone: using the block triangular variant instead of the block diagonal one, as done in Section 2.4.2, decreases significantly the number of iterations of GMRES(30) (134 against 315). Then, this example justifies the study led in this chapter to define a class of limited memory preconditioners for nonsymmetric matrices, although the original saddle point matrices in the sequence (3.19) are symmetric. The three variants of LMP_{ns} still perform better than using no second level preconditioner: even with $k = 5$, the number of iteration count is roughly divided by 4. We also note that using eigenvectors leads to the minimal number of iterations in all situations. To conclude this analysis, using Ritz or harmonic Ritz for S leads to a very similar convergence behaviour on this example. In practice, we have not obtained significant differences between both variants in all numerical experiments, and from now on, we present only the results given by the selection of Ritz vectors associated to the smallest in modulus Ritz values.

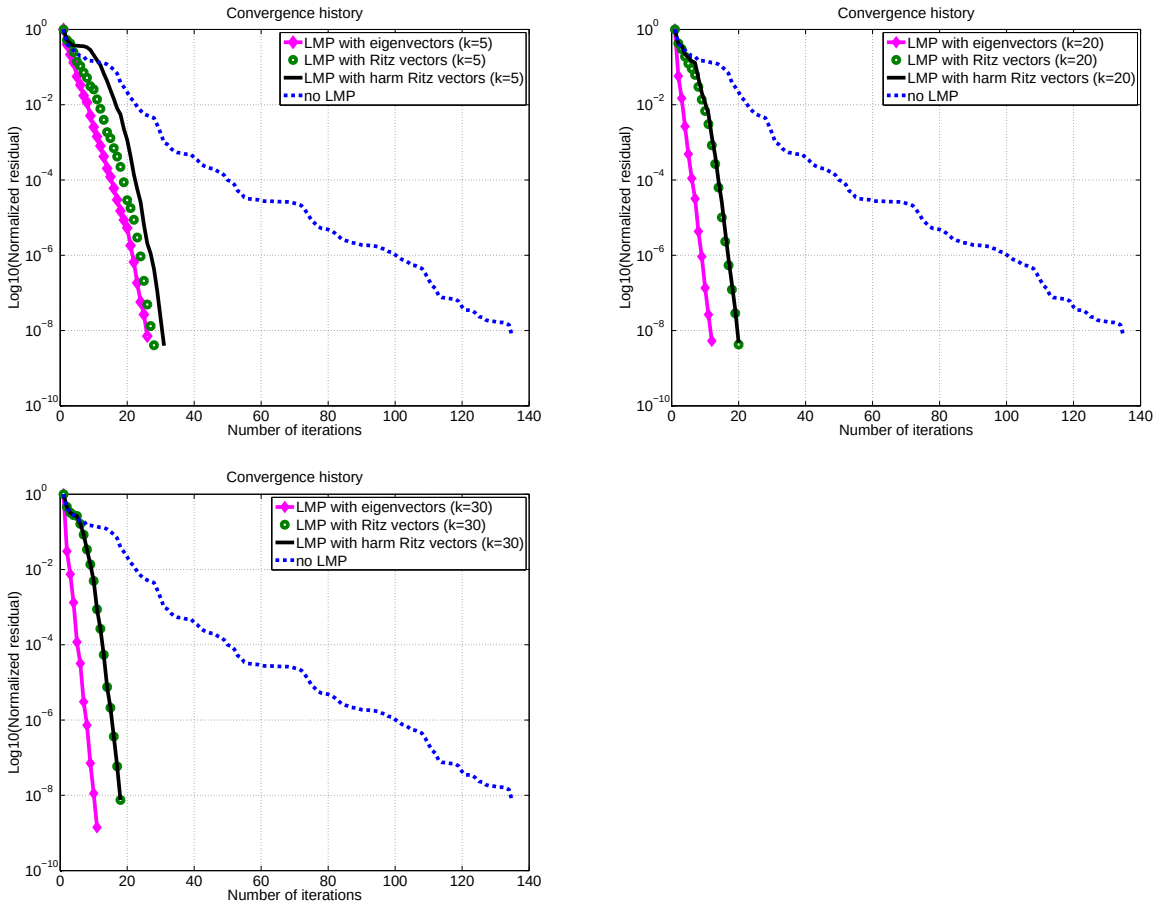

Figure 3.2: Mechanical bearing: convergence history of GMRES(30). Four preconditioning methods are compared: no second level preconditioner, limited memory preconditioners based on $k = 5$, 20 or 30 Ritz vectors, harmonic Ritz vectors and exact eigenvectors.

Secondly, we have compared the action of the LMP_{ns}, the deflation and the balancing preconditioner in this nonsymmetric case for the same matrices S. More precisely, we have applied the deflation and the balancing preconditioner as described in (3.15) and (3.17) respectively, and Figure 3.3 shows in particular the equivalence of the convergence histories with both methods (in agreement with Theorem 3.3.5). Moreover, the action of the limited memory preconditioner leads to similar behaviours.

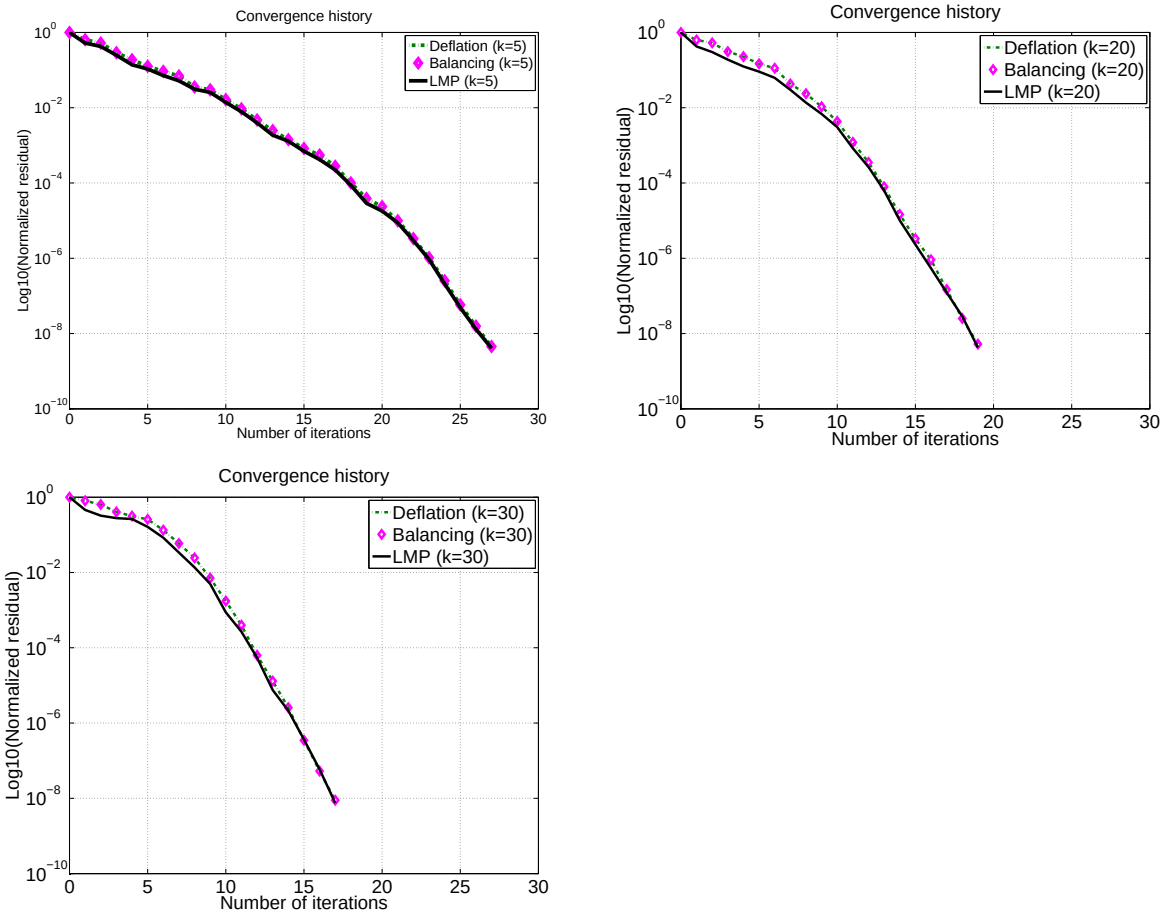

Figure 3.3: Mechanical bearing: convergence history of GMRES(30). Three methods are compared, all based on $k = 5$, 20 or 30 Ritz vectors: limited memory preconditioner, balancing preconditioner and deflation.

3.5.3 Containment building of a nuclear reactor

Now, we investigate the problem concerning the mechanical properties of the containment building of a nuclear reactor introduced in Section 2.4.3. The only change is related to the use of the block triangular augmented Lagrangian first level preconditioner, but with the same γ parameter and the same incomplete Cholesky factorization of $G_1 + \gamma B^T B$. This mechanical problem leads to a sequence of 4 nonsymmetric linear systems of the form (3.21), where the matrix is fixed.

Figure 3.4 illustrates the convergence history of GMRES(30) for the last three linear systems in the sequence ($i = 2, 3, 4$), with or without LMP_{ns}. As remarked in the

mechanical bearing test, the use of the block triangular first level preconditioner based on augmented Lagrangian method is more efficient than the block diagonal one. In this experiment, we consider also LMP$_{ns}$ with a varying number of Ritz vectors (k = 5, 20, 30). In addition, we show in Table 3.1 the cumulative iteration count over the last three linear systems, the CPU time and the memory requirements provided by PETSc, respectively. We note that the smallest number of iterations is always obtained when selecting a large value of Ritz vectors ($k = 30$). More precisely, in this case, the iteration count decrease is equal to 53%, associated to a gain in terms of CPU time of 51%. This satisfactory result comes at a price of a low increase in memory requirements (4%).

	No LMP$_{ns}$	LMP$_{ns}$, $k = 5$	LMP$_{ns}$, $k = 20$	LMP$_{ns}$, $k = 30$
Total iteration count	135	74	66	64
Iteration count decrease (%)	×	45	51	**53**
CPU time (sec)	75.6	42.9	40.2	36.9
CPU time decrease (%)	×	43	47	**51**
Memory (Mo)	6895	6950	7071	7172
Memory increase (%)	×	**0.8**	2.6	4

Table 3.1: Containment building: cumulative iteration count for the last three linear systems in the sequence, CPU time and memory requirements for different limited memory preconditioners. Case of $k = 5$, 20 or 30 Ritz vectors.

On the other hand, Table 3.2 summarizes the results obtained with the deflated GMRES(30) method. The gains are slightly better than those obtained with the LMP$_{ns}$, whether looking at iteration count or CPU time. In terms of memory requirements, we obtain approximately the same as the LMP$_{ns}$.

	No def.	def., $k = 5$	def., $k = 20$	def., $k = 30$
Total iteration count	135	74	64	59
Iteration count decrease (%)	×	45	53	**56**
CPU time (sec)	75.6	42.6	38	33.6
CPU time decrease (%)	×	43	50	**55.6**
Memory (Mo)	6895	6946	7068	7169
Memory increase (%)	×	**0.7**	2.5	3.9

Table 3.2: Containment building: cumulative iteration count for the last three linear systems in the sequence, CPU time and memory requirements for different deflation methods. Case of $k = 5$, 20 or 30 Ritz vectors.

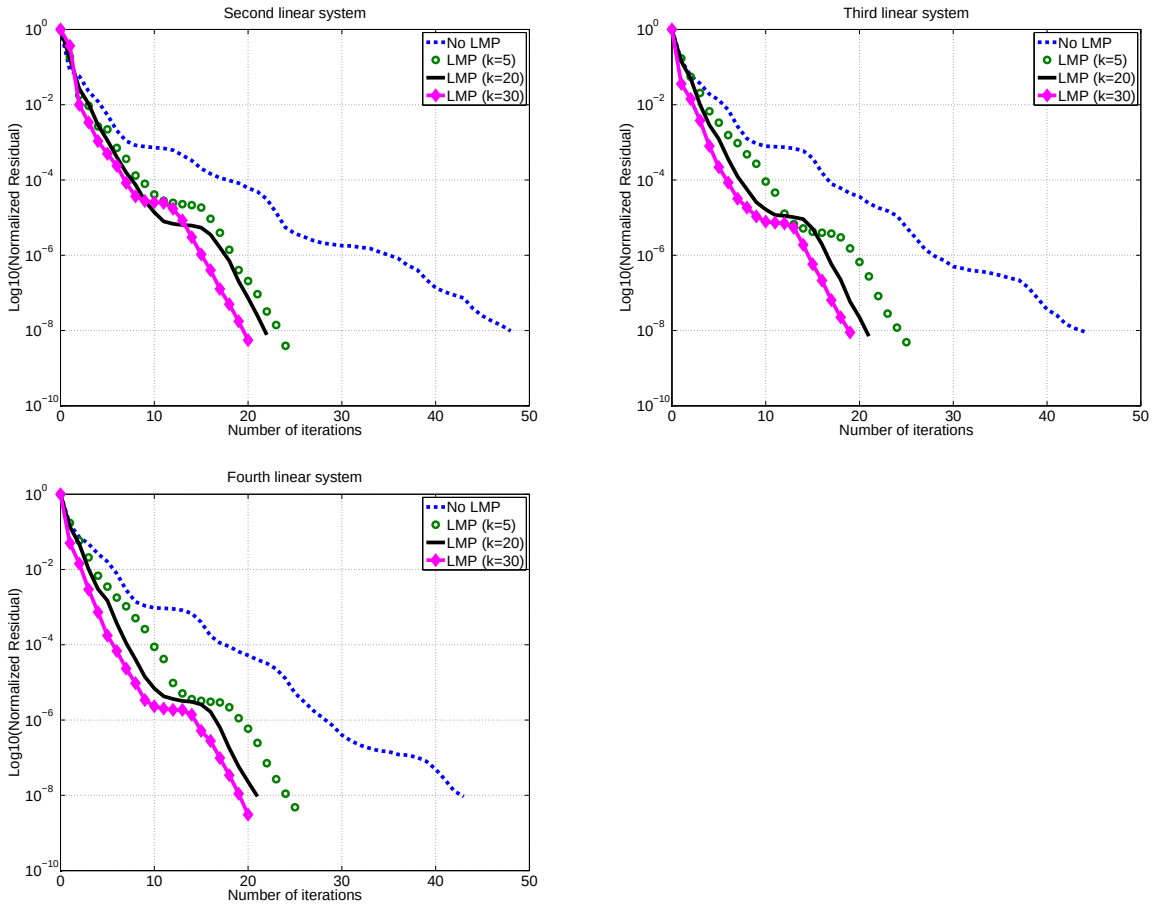

Figure 3.4: Containment building: convergence history of preconditioned GMRES(30) for the last three linear systems in the sequence. Case of limited memory preconditioners with $k = 5$, 20 or 30 Ritz vectors.

The matrix being fixed in the last two numerical experiments, we next show the action of the LMP_{ns} on sequences arising from Newton's method with slowly varying matrices. From now on, we combine the second level improving techniques with the *LDLT_SP* preconditioner, which is the default method in Code_Aster.

3.5.4 Shut down nuclear reactor cooling loop

This problem models the thermal fatigue of a shut down nuclear reactor cooling loop. The heat from the nuclear core is extracted by circulation of pressurized water in the primary circuit. When the reactor is shut down, the studied cooling loop allows to evacuate the heat of the primary circuit and the residual power of the nuclear fuel, using

cold water. Some oscillations of the temperature can occur where both cold and hot water are in contact and it is necessary to model the resistivity of the component to the thermal fatigue. The mesh of the studied material is illustrated in Figure 3.5.

Figure 3.5: Mesh of the shut down nuclear reactor cooling loop.

The discretized problem involves $N = 571281$ unknowns, with less than 1% of Lagrangian multipliers, corresponding to the essential dualized boundary conditions. Newton's method is employed because of the nonlinearity of the constitutive law of the structure (Chaboche elasto-visco-plastic law [88]) and leads to the sequence (3.22) of 67 linear systems. This solution is challenging, not only from its dimension, but also from the condition number of the saddle point matrices $\mathcal{K}_i$ which is of order of 10^{10} [3]. Indeed, it is known that this value can be notably related to the properties of the mesh; in particular, when there are strong spatial variations of the size of the mesh elements, or when some of them are flattened, the condition number increases [95]. The former case arises in this problem, at the intersection of both pipes (see Figure 3.5). Generally, when the condition number is moderate, the GMRES(30) method, preconditioned by the $LDLT\_SP$ technique, requires very few iterations to converge. With a large condition number, this iteration count is often larger (about 20 in this case), and we expect that a second level preconditioning technique, such as the LMP_{ns}, decreases this count.

We want to compare here the statistics related to the solution of the complete Newton's sequence, also including the $LDLT\_SP$ factorization step. Table 3.3 collects the results obtained with three different solution methods: without any second level improving technique, using the LMP_{ns} or the deflation for only $k = 5$. Indeed, in practice, the value $k = 5$ seems to be the better choice, since the number of GMRES(30) iterations with the first level preconditioner remains usually relatively low (i.e. less than 30) and a

3. This estimate has been obtained using MUMPS in double precision arithmetics as a direct solver on the first linear system.

larger k does not imply a significant additional gain. We first note that the use of either the limited memory preconditioner or the deflation method is very efficient in terms of iteration count with a gain of 53.3% and 58.3%, respectively. Concerning the CPU time, the decrease is largely in favour of the improvement preconditioning technique with a gain of 39.9%, against 9.2% for the deflation. This phenomena can be explained by the fact that the deflation operator P_D^i defined by (3.24) (as well as Q_D^i and R_D^i), needs to be updated at each new linear system, using the new matrix $\mathcal{A}_i$. However, according to Section 3.4, the orthonormalization process requires in particular 5 matrix-vector products by $\mathcal{A}_i$, which is roughly the computational cost of 5 iterations of GMRES(30) to solve $\mathcal{A}_i x_i = b_i$. In addition, the value 410 in Table 3.3 corresponds to the cumulative iteration count of the deflated GMRES(30) to solve the 67 linear systems, which means that each system requires in average about 6 iterations. Hence, the computational cost related to the update of P_D^i, Q_D^i and R_D^i is not negligible.

	No LMP$_{ns}$	LMP$_{ns}$, $k = 5$	Deflation, $k = 5$
Total iteration count	983	460	410
Iteration count decrease (%)	×	53.3	**58.3**
CPU time (sec)	961	578	873
CPU time decrease (%)	×	**39.9**	9.2
Memory (Mo)	14074	14117	14095
Memory increase (%)	×	0.03	**0.014**

Table 3.3: Shut down nuclear reactor cooling loop: cumulative iteration count over the complete Newton's sequence, CPU time and memory requirements for limited memory preconditioner and deflation with $k = 5$ Ritz vectors.

We take the opportunity in this section to illustrate the scalability of the limited memory preconditioners developed in this manuscript. The solution of this nonlinear problem has been computed with different numbers of processors and Figure 3.6 presents the speedup curves obtained with or without LMP$_{ns}$. The speedup factor is defined as

$$S(p) = \frac{T(1)}{T(p)},$$

where $T(p)$ corresponds to the CPU time of the method executed on p processors. Then, Figure 3.6 illustrates the ratio of the sequential CPU time to the parallel CPU time, in function of the number of processors. We note that both curves are relatively far from the ideal situation $S(p) = p$, but they are close to each other and we conclude that the implementation of the limited memory preconditioner does not deteriorate the

degree of scalability of the first level preconditioner ($LDLT\_SP$ here). We note that the parallelism of the $\text{LMP}_{\pm}$ is not illustrated in the manuscript but the associated operations are similar to the LMP_{ns}.

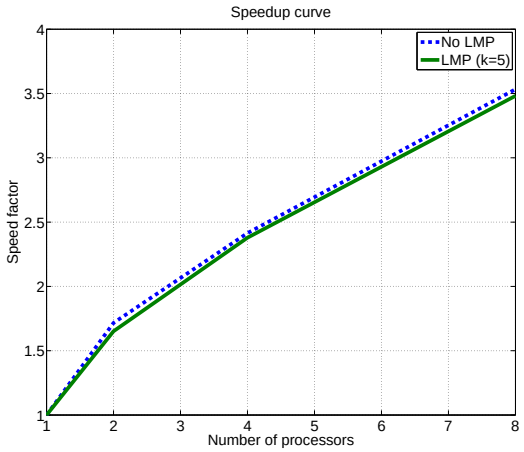

Figure 3.6: Shut down nuclear reactor cooling loop: speedup curve

3.5.5 Snap hook

The last problem is a reference performance test of Code_Aster [94]. The structure is a hollow snap hook, subject to an internal pressure. We consider a nonlinear constitutive law of the material (elastoplastic law of Von Mises with isotropic hardening), and we obtain from the Newton's method a sequence of 34 linear systems of the form (3.22) with multiple right and left-hand sides. The number of unknowns is $N = 504012$ with about 1% of Lagrange multipliers corresponding to essential boundary conditions. The condition number [4] of the saddle point matrices is large, about 10^{11}, and we will see that the action of the $LDLT\_SP$ can be improved using in particular the LMP_{ns}. The condition number can also been explained by the quality of the mesh: we guess in Figure 3.7 that the mesh elements on the curvatures of the snap hook have very different shapes from the others. Another important parameter has to be taken into account: in Code_Aster, the $LDLT\_SP$ preconditioner is computed from the matrix $\mathcal{A}_1$ in (3.22) and used for the successive linear systems. However, if the Krylov subspace method requires more than 30 iterations to converge for a certain system $\mathcal{A}_i x_i = b_i$ [5], a new $LDLT\_SP$ factorization is performed for the following matrix $\mathcal{K}_{i+1}$. Indeed, we consider that the effect of the preconditioner is degraded and we prefer to recompute it.

4. See 3.
5. 30 is the default value in Code_Aster.

In such a case, when a LMP_{ns} is used, H_{ns} is discarded after the solution of the i-th linear system, and a new one is defined at the end of the solution of $\mathcal{A}_{i+1}x_{i+1} = b_{i+1}$ with new Ritz vectors. This strategy is also adapted in the deflation method. We emphasize that this situation did not happen in the previous numerical experiment.

Figure 3.7: Mesh of the snap hook.

Table 3.4 collects the statistics of three solution methods over the complete Newton's sequence, where the CPU time includes the $LDLT\_SP$ factorization steps. Three methods are presented: no second level improvement technique, with LMP_{ns} or using the deflation for $k = 5$. We note that using a second level strategy is very efficient in terms of iteration count, with a gain of 39.5% and 40.3% for the preconditioning and deflating techniques, respectively. Concerning the number of $LDLT\_SP$ factorizations, both improvement methods are interesting, since they are able to decrease this number from 10 to 6. Finally, cumulating the different gains, the LMP_{ns} is more efficient in terms of CPU time, with a decrease of 34.3%. The same arguments as in Section 3.5.4 justify the fact that the deflation method is less efficient in terms of CPU time. Nevertheless, the cost involved to update the deflation operators represents a smaller part of the total cost in this case. Table 3.4 finally shows that the memory overcost in negligible with both methods.

	No LMP$_{ns}$	LMP$_{ns}$, $k = 5$	Deflation, $k = 5$
Total iteration count	1058	641	**632**
Iteration count decrease (%)	×	39.5	**40.3**
$LDLT\_SP$ factorizations	10	**6**	**6**
CPU time (sec)	534	340	384
CPU time decrease (%)	×	**34.3**	28.1
Memory (Mo)	3984	4015	3999
Memory increase (%)	×	0.8	**0.4**

Table 3.4: Snap hook: cumulative iteration count over the complete Newton's sequence, number of required $LDLT\_SP$ factorizations, CPU time and memory requirements for limited memory preconditioner and deflation with $k = 5$ Ritz vectors.

3.6 Conclusions

In this chapter, we have proposed a class of limited memory preconditioners adapted to the solution of nonsymmetric linear systems. This preconditioner can be interpreted as a block generalization of update formulas proposed in both EN and Broyden optimization methods.

First, we have shown that the spectrum of the preconditioned operator contains the eigenvalue 1 (with multiplicity at least k) and the remaining part of the spectrum has been characterized. Then, this new class of preconditioners has been compared with existing methods: the deflation and the abstract balancing preconditioning technique. The three associated spectra are similar (except k possibly different eigenvalues). Moreover, we have proved that the use of GMRES on the deflated system, or right preconditioned by the abstract balancing preconditioner give identical iterates, provided that appropriate initial guesses and right-hand sides are chosen.

The numerical part of this chapter deals with solid mechanics problems leading to sequences of symmetric saddle point linear systems, as tested in Chapter 2. Here, any first level preconditioner can be used here, making the successive preconditioned operators nonsymmetric. Numerical experiments emphasize the efficiency of the class of LMP$_{ns}$. On the one hand, using a block upper triangular first level preconditioner, a saving of up to 51% in terms of computational time is obtained on a large-scale application. On the other hand, if we define the first level preconditioner as a complete factorization in single precision arithmetics with MUMPS, we obtain a gain of 39.9% on a nuclear safety analysis corresponding to the study of a shut down reactor cooling loop. Simultaneously, we have compared the action of the LMP$_{ns}$ with the action of the deflated GMRES method. In terms of iteration count, the gains are close, even if the deflation

method gives slightly better results on the large-scale systems illustrated here. However, when the left-hand sides change during the sequence, the deflation operators need to be updated, which makes it less competitive in terms of computational time. Lastly, the scalability of the limited memory preconditioners has been illustrated on one large-scale example in this chapter.

Conclusions & Perspectives

Conclusions

The main goal of this thesis is to provide efficient methods to improve the convergence rate of Krylov subspace methods for the solution of sequences of linear systems, arising in solid mechanics. Simultaneously, we aim at implementing these methods in the open source Code_Aster software, a finite element code developed at EDF, which is used as a simulation tool by the engineering departments to produce notably safety analysis.

Chapter 1 has introduced and detailed the context of this research. First, we have presented the treatment of nonlinearities in solid mechanics, as well as different types of imposed conditions. In several industrial software like Code_Aster, these problems lead to sequences of linear systems with saddle point structure, considered to be symmetric in this manuscript. With this in mind, we have introduced the necessary mathematical background related to the Krylov subspace methods, which are the methods of choice for such a purpose, as well as state-of-art solution methods for saddle point linear systems.

In this framework, the thesis has contributed to the research area related to algebraic preconditioning strategies for the GMRES Krylov subspace method, especially for the solution of sequences of linear systems with fixed or slowly varying left-hand sides. Two methods have been proposed and studied, both based on updating formulas issued from the numerical optimization literature. They respectively address the following challenges:

(i) Extending to the symmetric indefinite case the class of limited memory preconditioners [46], initially developed for the solution of symmetric positive definite linear systems.

(ii) Proposing another class of limited memory preconditioners, able to handle non-symmetric systems.

The challenge (i) has been addressed in Chapter 2, where the definition proposed for the new class of preconditioners (called LMP$_\pm$) is similar to the one given in [46]. More precisely, both are defined by the same formula, which can be seen as a block generalization of the BFGS updating formula for quadratic problems, and involve k linear independent vectors, composing the columns of a matrix S. However, the matrix S^TAS has to be nonsingular, when the matrix A of the linear system is symmetric indefinite. Given this definition, several contributions have been proposed. First, the spectrum of the preconditioned operator has been analysed for any matrix S satisfying the assumptions given above. This characterization has revealed that the eigenvalues are all real-valued, with at least k equal to 1. Then, a discussion has been led to obtain the sign of these eigenvalues and the inertia of the LMP$_\pm$. The second contribution has concerned the nonexpansion of the spectrum of the preconditioned matrix relatively to the original one. Contrary to the symmetric positive definite case, this property is not naturally inherited for any LMP$_\pm$, but a similar result is obtained when the columns of S are prior projected onto the invariant subspaces associated with the eigenvalues of the original matrix in the open right and left-half plane, respectively. This latter case is generally computationally too expensive in practice, but reveals that selecting columns of S spanning an approximate invariant subspace is relevant. In this sense, we have contributed in studying the Ritz-LMP variant, where Ritz information is used to determine the matrix S.
The LMP$_\pm$ has been developed into Code_Aster, via the PETSc library, in combination with the GMRES method. Thanks to this code, the efficiency of this preconditioning technique has been illustrated on several problems in solid mechanics, leading to sequences of linear systems with saddle point structure. In fact, the LMP$_\pm$ is particularly well suited to improve the effect of an existing first-level preconditioner. In this case, this latter has to be symmetric positive definite, in order to apply the LMP$_\pm$ on a symmetric operator. In this chapter, a block diagonal first-level preconditioner has been considered. A first small-scale problem has revealed a huge gain in terms of GMRES iteration count and has strengthened our belief in selecting Ritz vectors to define the LMP$_\pm$. Then, two large-scale real-life problems have been studied, leading to significant gains in terms of computational time, i.e., up to 41% and 17.2%, whether the matrix is fixed or slowly varying, respectively. Finally, all these results have been obtained at a negligible memory requirement overcost.

The challenge (ii) has been addressed in Chapter 3, where nonsymmetric linear systems are considered. Beyond the larger scope of possible applications, this approach has allowed us to consider any first-level preconditioner for the sequences of symmetric

saddle point linear systems studied in this thesis. We note that this new preconditioning technique (called LMP_{ns}) can also be seen as a block generalization of updating formulas issued from the optimization literature (from variants of Broyden and EN methods, precisely), involving a matrix S whose k columns are simply assumed to be linearly independent. The theoretical part of this chapter has offered three main contributions. The first one has concerned the spectral characterization of the application of a LMP_{ns} to a matrix A, clustering notably at least k eigenvalues at 1. Then, the study aimed at positioning and comparing the LMP_{ns}, as well as the $\text{LMP}_{\pm}$ in the symmetric case, with two existing methods: the deflation and the abstract balancing preconditioner. The second contribution in this analysis has concerned the spectral comparison, showing that all these methods lead to similar spectra of the associated deflated or preconditioned operators. The third theoretical contribution has dealt with the comparison of the GMRES iterates provided by the different methods. More precisely, we have shown that the convergence histories are similar using either the deflation method or the abstract balancing preconditioner as a right preconditioner on an adapted linear system with an appropriate initial guess. This result, illustrated on a numerical experiment, also concerns the $\text{LMP}_{\pm}$ in the symmetric indefinite case, since it is equal to the abstract balancing preconditioner in this context.

After presenting a possible implementation of the LMP_{ns}, as well as the associated deflation method, Chapter 3 has illustrated the efficiency of these improvement techniques on sequences of saddle point linear systems combined with two different first-level preconditioners. These numerical tests have been obtained with a code developed in Code_Aster and PETSc. As in the symmetric indefinite case, the strategy to define the LMP_{ns} or the deflated operators, based on Ritz information associated to smallest in modulus Ritz values, gives attractive results. Globally, we have obtained significant gains, both in terms of GMRES iteration count and of computational time, using the LMP_{ns} on large-scale problems. In particular, we have tested this method on linear systems with large condition numbers, using a single precision arithmetic factorization as a first-level preconditioner. In these cases, the gain in CPU time can reach up to 39.9%, which is very significant from an industrial point of view. Moreover, the associated increase in terms of memory requirements remains negligible. Beyond the efficiency of the LMP_{ns}, we have also obtained good results using the deflation. However, due to the fact that the deflated operators need to be recomputed before the solution of each linear system (if the matrix changes), the decrease is found to be less impressive in terms of computational time. Finally, an important point has been illustrated in the numerical section of Chapter 3, concerning the scalability of the class of limited memory preconditioners proposed

in this thesis. Indeed, we have shown, on a large-scale problem, that this improvement technique allows us to preserve the degree of scalability of the first-level preconditioner.

Perspectives

Several extensions to the present research can be mentioned. We split these directions in two parts, dealing first with the both $\text{LMP}_{\pm}$ and LMP_{ns} and then with the preconditioning strategies in Code_Aster.

LMP

We give three future directions related to the application of both $\text{LMP}_{\pm}$ and LMP_{ns}. First, we have emphasized in this thesis the role of the selected information as columns of the matrix S in (2.6) and (3.4). In practice, the Ritz vectors corresponding to the smallest in modulus Ritz values have performed best in all the numerical experiments. In fact, although not illustrated in the numerical sections of the manuscript, we have also tested other strategies based for instance on the smallest approximation errors of the Ritz pairs (defined in (1.15)), but the results were less attractive in terms of computational time. It is worth mentioning that other authors have observed a similar behaviour in a different context: we can cite for instance the paper review [38], dedicated to recycling Krylov subspace methods to solve sequences of linear systems, with application on nonlinear Schrödinger equations. As an example of an alternative, we can refer the reader to [45], where the authors propose a selective reuse of Krylov subspaces for the solution of sequences of symmetric positive definite systems in solid mechanics.
Another point would merit further discussion: the update of the $\text{LMP}_{\pm}$ and the LMP_{ns} during the solution of the sequence of linear systems. More precisely, we have considered in the numerical experiments that the preconditioner is defined at the end of the solution of the first linear system and is kept fixed. Besides, this strategy has shown to be more efficient than deflation on real-life numerical tests. Nevertheless, when the effect of the LMP deteriorates during the sequence, it would be interesting to consider a new improvement technique. A first analysis has been conducted during this thesis, based for instance on the accumulation of several LMP_{ns}, or on the choice of new columns in S at the end of each linear system. These studies are not enough achieved to be presented in this manuscript. Concerning the update of the columns of S, we can refer the reader to [38, 83].
Thirdly, the LMP_{ns} can obviously be used as a second-level preconditioner for nonsymmetric problems in solid mechanics, as well as in other fields of application.

Preconditioning

This thesis has provided prospects about preconditioning strategies in Code_Aster. Indeed, as mentioned earlier, the current preconditioners available in this software do not take into account the saddle point structure of the systems. During this thesis, we have developed an interface with the PETSc library to take into account this block structure. In particular, we have illustrated the effect of two block preconditioners on large-scale numerical experiments, where the (1,1) block is handled by an incomplete Cholesky factorization computed by PETSc. An interesting approach would be to use MUMPS, known to be an efficient library for sparse direct solvers, to factorize this block. First tests have been performed, providing attractive results. Furthermore, since this factorization is related to preconditioning, using MUMPS in single arithmetic precision could be more efficient in terms of memory requirements and computational time. This latter approach would necessitate to develop in PETSc the possibility to handle different arithmetics, a feature which is not currently available.

Finally, we have focused on the solution of linear systems issued from Newton's method. Another interesting point to explore within Code_Aster would be related to the nonlinear aspect. Currently, an alternative to the Newton's method is available in Code_Aster, based on the adaptation of the stopping criterion of the solution of each linear system [105, 59]. Furthermore, the Jacobian-Free Newton-Krylov [63] approach seems attractive. The main idea behind this method is that the Krylov subspace methods do not require the knowledge of the matrix but just its action on a vector. We could obtain significant additional gains, since all the left-hand sides would not need to be built (just those from which a preconditioner would be computed). We further note that the classes of limited memory preconditioners proposed in this dissertation could be used in combination with this nonlinear solver as well.

Appendix A

Description of the code

In this appendix, we succinctly describe the routines developed in both the Code_Aster software and the PETSc library, related to the different preconditioning techniques studied in this manuscript. More precisely, we focus on the methods introduced in Chapter 3, which provide significant gains in terms of computational time and do not require any restriction about the choice of the first-level preconditioner. We emphasize that the implementation has been done in a parallel framework, as illustrated in Figure 3.6. Finally, we notify the reader that this appendix refers to several routines specific to PETSc [8].

Remark 14. The implementation of the methods analysed in Chapter 2 is very similar to what follows.

The main idea is to use the PETSc library as an iterative solver for the solution of each linear system in the sequence, and more specifically the GMRES method, as justified in Chapters 2 and 3. This approach is already available in Code_Aster, thanks to an interface developed in Fortran 90. It is worth mentioning that the nonlinear process is handled by Code_Aster and that PETSc just handles the solution of the successive linear systems. The appendix is split in two main parts. The first one gives information on how the saddle point structure of the systems is taken into account in the code; this has been particularly developed in order to define the block upper triangular first-level preconditioner in Sections 3.5.2 and 3.5.3. The second part illustrates the developments carried out both in Code_Aster and PETSc to implement the LMP_{ns} method.

A.1 Block preconditioners for saddle point systems

Currently, the different preconditioning techniques available in Code_Aster treat the global problem and unfortunately do not take into account the saddle point structure of

the systems. Hence, we have developed an interface with PETSc to recover the different blocks of the matrices. Let us focus on the linear system

$$\mathcal{K}x = b \Longleftrightarrow \begin{pmatrix} G & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ \lambda \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix}. \tag{A.1}$$

As mentioned in Section 1.3.2.1, this system is automatically modified, thanks to the "double Lagrange" method, into

$$\begin{pmatrix} G & B^T & B^T \\ B & -I_m & I_m \\ B & I_m & -I_m \end{pmatrix} \begin{pmatrix} u \\ \lambda_1 \\ \lambda_2 \end{pmatrix} = \begin{pmatrix} f \\ g \\ g \end{pmatrix}, \text{ with } \begin{cases} \lambda_1 = \lambda_2 \\ \lambda = \lambda_1 + \lambda_2 \end{cases}. \tag{A.2}$$

Remark 15. For the sake of clarity in this presentation, we consider that the α coefficient used to scale the different blocks in (1.22) is equal to 1.

If the PETSc library is chosen by the user to solve this linear system, both the left and right-hand sides are duplicated in the PETSc format. From the numbering corresponding to the problem unknowns, specific to Code_Aster, we then define some `Index Set` (PETSc objects) to recover:

- the vectors f and g associated to the physical unknowns and the Lagrange multipliers, respectively. To do so, we call the `VecScatterBegin` and `VecScatterEnd` routines, which recover the required vectors from the different processors. We further note that the initial guess used in the Krylov subspace methods in Code_Aster is always zero. Hence, it is not necessary to apply this step on this vector.

- the G and B blocks corresponding to the stiffness and the constraint matrices, respectively. They are recovered with the PETSc routine called `MatGetSubMatrix`.

Since all the necessary data are now retrieved, we can solve the linear system in the form (A.1), i.e. without involving the "double Lagrange" approach. This can be done using the `MATSHELL` notion in PETSc, which allows the user to define his/her own matrix-vector product. Now, it is also possible to define the following block upper triangular preconditioner (used as a first-level preconditioner in Section 3.5):

$$\widetilde{\mathcal{M}}_t = \begin{pmatrix} LL^T & 2B^T \\ 0 & -I_m \end{pmatrix}, \tag{A.3}$$

where $LL^T \approx G + B^T B$. In practice, this incomplete Cholesky decomposition is computed by the inhouse PETSc method called `PCICC` and used in combination with a RCM

reordering technique. Once this factorization has been obtained, we can define a routine to apply the preconditioner $\widetilde{\mathcal{M}}_t$, using a `PCSHELL`, which is based on the same idea as the `MATSHELL` notion. Finally, the linear system (A.1) is solved through the PETSc routine called `KSPSolve`. When the solution is computed, it is copied using the initial numbering of Code_Aster, thanks to the `Index Set` previously defined.

Remark 16. When a sequence of linear systems has to be solved, we need to recover and substitute the blocks of the left and right-hand sides for each system. However, the first-level preconditioner $\widetilde{\mathcal{M}}_t$ is computed from the first saddle point matrix and remains fixed all along the sequence.

A.2 Limited memory preconditioners

We focus in this section on the implementation of the LMP$_{ns}$ class, suited to solve nonsingular problems (see Chapter 3). It has been decided that this nonsymmetric formulation would be delivered in a future version of Code_Aster. Generally speaking, we want to use GMRES(restart) to solve the following sequence:

$$\mathcal{M}\mathcal{A}_1 x_1 = \mathcal{M} b_1 \quad \text{and} \quad \left\{ \begin{array}{ll} \mathcal{M}\mathcal{A}_i H_{ns} \tilde{x}_i = b_i & \\ x_i = H_{ns} \tilde{x}_i & \end{array} \right. \quad i = 2, \cdots, I, \tag{A.4}$$

where $\mathcal{M}$ is the first-level preconditioner and H_{ns} is the LMP$_{ns}$. We next present the routines developed in PETSc and Code_Aster, successively.

A.2.1 Development in PETSc

It is worth mentioning that the definition of H_{ns} is closely related to the selection of the columns of the matrix S in (3.4). In this manuscript, the strategy is based on Ritz vectors (or harmonic Ritz vectors), recovered after the solution of the first linear system of the sequence (A.4). However, no PETSc routine allows to obtain such information. Thus, we have developed this feature. All our contributions in PETSc (developed in C) are summarized below.

A.2.1.1 Approximations of eigenpairs based on (harmonic) Ritz pairs

This functionality has been developed in two steps, each step being related to a specific routine. We emphasize that these routines are more generic. Actually, they allow the user to obtain spectral approximations and can be used for other purposes than the

definition of a LMP$_{ns}$.

First, let us recall that our strategy is based on the computation of (harmonic) Ritz vectors from the Hessenberg matrix obtained during the last complete cycle of GMRES(restart). Then, the `KSPSetComputeRitz` routine allows us to save (or not) this matrix. The routine is called in Fortran as

`KSPSetComputeRitz(ksp,flag,ierr)`, where

- `ksp` is an input PETSc object of `KSP` type, which manages the parameters of the iterative solver;
- `flag` is an input boolean equal to 1 if the user wants to save the last complete Hessenberg matrix. Otherwise, the value is set to 0;
- `ierr` is a PETSc datatype used for return error code.

We further note that if no complete cycle has been performed, we save the Hessenberg matrix obtained at the end of the solution. Moreover, this routine needs to be called before the `KSPSetUp` routine, setting up the internal data structures for the use of the Krylov subspace method.

Secondly, the `KSPComputeRitz` routine computes the required (harmonic) Ritz vectors, after the solution phase has been handled in GMRES(restart). We next detail all the parameters of this routine called in Fortran, where I and O correspond to input and output parameters, respectively.

`KSPComputeRitz(ksp,ritz,small,S,tetar,tetai,nbrit,ierr)`, where

- `ksp` (I) is a PETSc object of `KSP` type, which manages the parameters of the iterative solver;
- `ritz` (I) is a boolean equal to 1 to recover the Ritz pairs and 0 for the harmonic ones;
- `small` (I) is a boolean equal to 1 to recover the Ritz pairs corresponding to the smallest values in modulus, and 0 for the largest ones;
- `S` (O) is a multidimensional PETSc vector object containing the selected vectors;

- `tetar` (O) is an array containing the real part of the selected (harmonic) Ritz values;

- `tetai` (O) is an array containing the imaginary part of the selected (harmonic) Ritz values;

- `nbrit` (I/O) is an integer, whose input and output values are equal to the number of required and recovered pairs, respectively;

- `ierr` (O) is a PETSc datatype used for return error code.

We emphasize that `nbrit` can have different input and output values, since the (harmonic) Ritz pairs are possibly complex-valued (see the discussion in Section 3.4.1). Actually, in such a case, the routine selects the complex (harmonic) Ritz value and its conjugate, and two of the columns of S are equal to the real and the imaginary parts of the associated vectors. Finally, we stress that this functionality is only available in the GMRES context.

A.2.1.2 Preconditioning

Using the routines introduced above, we can define a LMP_{ns} from Ritz or harmonic Ritz vectors. However, we use both left and right preconditioners when solving the sequence (A.4). It is not possible in PETSc to apply both left and right preconditioners using the `PCSHELL` notion (i.e. when the user defines his/her own application routines for these preconditioners). Thus, we have developed this new feature, which is available by calling in Fortran the new following functions:

`KSPSetShellSetApplyLeft(pc,pcapl,ierr)` and `KSPSetShellSetApplyRight(pc,pcapr,ierr)`, where

- `pc` (I) is a PETSc object of `PC` type, which manages the parameters of the preconditioner;

- `pcapl` or `pcapr` (I) is the name of the routine where, the application of the preconditioner on a vector is defined;

- `ierr` (O) is a PETSc datatype used for return error code.

A.2.2 Development in Code_Aster

Now, we have detailed almost all the necessary information to solve the sequence (A.4) in Code_Aster. The development performed in this software is based on several PETSc routines (including the new ones presented before). However, another central step remains to be detailed. Indeed, the `KSPSetComputeRitz` routine allows the user to recover the matrix S whose columns are defined from the Ritz or harmonic Ritz vectors (at least the real and imaginary parts). In order to apply the LMP$_{ns}$, as proposed in Section 3.4.2, we need to implement the orthonormalization process detailed in Algorithm 12. We have decided to do it inside Code_Aster, in order to let the contribution related to PETSc as general as possible. This routine is not detailed in this appendix, but follows the successive steps of Algorithm 12. To conclude, the following code aims at presenting the main steps of a simplified version of the code developed in Code_Aster (in Fortran 90), related to the use of the LMP method to solve a sequence of linear systems. We note that this program does not illustrate the development done to take into account the saddle point structure of the systems.

```
! PROGRAM TO SOLVE A SEQUENCE USING THE LMP METHOD

! inclusion of routines
#include "flp.h"      ! routine to apply the first-level preconditioner
#include "lmp.h"      ! routine to apply the LMP

! declaration of variables
KSP         :: ksp                  ! variable for the iterative solver
PC          :: pc                   ! variable for the preconditioner
integer     :: i,nbrit
Vec         :: x,b                  ! solution and right-hand side
Vec         :: S(k),X(k),Y(k)       ! multidimensional vectors for the LMP
PetscReal   :: tr(k),ti(k)          ! multidimensional arrays for the real and
                                    ! imaginary parts of the Ritz values
```

```
! select GMRES as the Krylov subpace method
call KSPSetType(ksp, KSPGMRES, ierr)
! set several parameters of GMRES (stopping criterion, restart ...)
...
! loop on the linear systems
if (i.eq.1) then
!          define the left and right-hand sides...
           ...
!          define the left preconditioner using a PCSHELL
           call KSPSetPCSide(pc,PC_LEFT, ierr)
           call KSPShellSetApply(pc, flp, ierr)
!          save the last complete Hessenberg matrix
           call KSPSetComputeRitz(ksp,1, ierr)
!          solution of the linear system
           call KSPSolve(ksp,x,b, ierr)
!          computation of approximate spectral information
           call KSPComputeRitz(ksp,1,1,S, tetar, tetai, nbrit, ierr)
!          orthonormalization process to obtain H = I + YX^T
           call orthoritz(S,X,Y)
else
!          define the left and right-hand sides...
           ...
!          define the left and right preconditioners using PCSHELL
           call KSPSetPCSide(pc,PC_SYMMETRIC, ierr)
           call KSPShellSetApplyLeft(pc, flp, ierr)
           call KSPShellSetApplyRight(pc, lmp, ierr)
           call KSPSetComputeRitz(ksp,0, ierr)
!          solution of the linear system
           call KSPSolve(ksp,x,b, ierr)
end
end
```

Bibliography

[1] Code_Aster. http://www.code-aster.org.

[2] MUMPS. http://mumps.enseeiht.fr.

[3] PETSc. http://www.mcs.anl.gov/petsc.

[4] SLEPc. http://slepc.upv.es.

[5] M. Abbas. Algorithme non linéaire quasi-statique (opérateur STAT_NON_LINE). Technical Report, EDF R&D, AMA, 2013. R5.03.01.

[6] D. Al-Akhrass. *Méthodes éléments finis mixtes robustes pour gérer l'incompressibilité en grandes déformations dans un cadre industriel.* PhD thesis, Saint-Etienne, EMSE, 2014.

[7] Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. A. van der Vorst. *Templates for the solution of algebraic eigenvalue problems: a practical guide*, volume 11. SIAM, 2000.

[8] S. Balay, S. Abhyankar, M. F. Adams, J. Brown, P. Brune, K. Buschelman, L. Dalcin, V. Eijkhout, W. D. Gropp, D. Kaushik, M. G. Knepley, L. C. McInnes, K. Rupp, B. F. Smith, S. Zampini, and H. Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 3.6, Argonne National Laboratory, 2015.

[9] F. Ben Belgacem, P. Hild, and P. Laborde. Extension of the mortar finite element method to a variational inequality modeling unilateral contact. *Mathematical Models and Methods in Applied Sciences*, 9(02):287–303, 1999.

[10] S. Bellavia, D. Bertaccini, and B. Morini. Nonsymmetric preconditioner updates in Newton-Krylov methods for nonlinear systems. *SIAM J. Sci. Comput.*, 33(5):2595–2619, 2011.

[11] M. Benzi. Preconditioning techniques for large linear systems: a survey. *J. Comp. Phys.*, 182(2):418–477, 2002.

[12] M. Benzi and D. Bertaccini. Approximate inverse preconditioning for shifted linear systems. *BIT*, 43(2):231–244, 2003.

[13] M. Benzi and G. H. Golub. A preconditioner for generalized saddle point problems. *SIAM J. Matrix Anal. Appl.*, 26(1):20–41, 2004.

[14] M. Benzi, G. H. Golub, and J. Liesen. Numerical solution of saddle point problems. *Acta Numer.*, 14(1):1–137, 2005.

[15] M. Benzi and V. Simoncini. On the eigenvalues of a class of saddle point matrices. *Numer. Math.*, 103(2):173–196, 2006.

[16] M. Benzi and M. Tuma. A comparative study of sparse approximate inverse preconditioners. *Appl. Num. Math.*, 30(2):305–340, 1999.

[17] L. Bergamaschi, J. Gondzio, M. Venturin, and G. Zilli. Inexact constraint preconditioners for linear systems arising in interior point methods. *Comput. Optim. Appl.*, 36(2-3):137–147, 2007.

[18] J. Bloch and S. Heybrock. A nested Krylov subspace method to compute the sign function of large complex matrices. *Comp. Phys. Comm.*, 182(4):878–889, 2011.

[19] O. Boiteau. Mot-clé solveur. Technical Report, EDF R&D, AMA, 2014. U4.03.05.

[20] J. Bonet and R. D. Wood. *Nonlinear continuum mechanics for finite element analysis*. Cambridge University Press, 1997.

[21] M. Bonnet and A. Frangi. *Analyse des solides déformables par la méthode des éléments finis*. Editions Ecole Polytechnique, 2006.

[22] J. H. Bramble and J. E. Pasciak. A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems. *Math. Comp.*, 50(181):1–17, 1988.

[23] C. G. Broyden. A new method of solving nonlinear simultaneous equations. *The Computer Journal*, 12(1):94–99, 1969.

[24] S. L. Campbell, I. CF. Ipsen, C. T. Kelley, and C. D. Meyer. GMRES and the minimal polynomial. *BIT*, 36(4):664–675, 1996.

[25] A. Chapman and Y. Saad. Deflated and augmented Krylov subspace techniques. *Numer. Linear Algebra Appl.*, 4(1):43–66, 1997.

[26] M. Corus. Élimination des conditions aux limites dualisées. Technical Report, EDF R&D, AMA, 2014. R3.03.05.

[27] T. A. Davis. *Direct methods for sparse linear systems*, volume 2. SIAM, 2006.

[28] H. S. Dollar, N. IM. Gould, M. Stoll, and A. J. Wathen. Preconditioning saddle-point systems with applications in optimization. *SIAM J. Sci. Comput.*, 32(1):249–270, 2010.

[29] Z. Dostál. Conjugate gradient method with preconditioning by projector. *Int J. Comput. Math.*, 23(3):315–323, 1988.

[30] N. Dyn and W. E. Ferguson. The numerical solution of equality constrained quadratic programming problems. *Math. Comp.*, 41(163):165–170, 1983.

[31] T. Eirola and O. Nevanlinna. Accelerating with rank-one updates. *Linear Algebra Appl.*, 121:511–520, 1989.

[32] H. C. Elman, D. J. Silvester, and A. J. Wathen. *Finite elements and fast iterative solvers: with applications in incompressible fluid dynamics.* Oxford University Press, 2005.

[33] Y. Erlangga and R. Nabben. Deflation and balancing preconditioners for Krylov subspace methods applied to nonsymmetric matrices. *SIAM J. Matrix Anal. Appl.*, 30(2):684–699, 2008.

[34] G. Fasano and M. Roma. A novel class of approximate inverse preconditioners for large positive definite linear systems in optimization. *Comput. Optim. Appl.*, pages 1–31, 2015.

[35] A. Fortin and A. Garon. Les éléments finis: de la théorie à la pratique. *Université Laval, Canada*, 2011.

[36] A. Frommer, S. Güttel, and M. Schweitzer. Efficient and stable Arnoldi restarts for matrix functions based on quadrature. *SIAM J. Matrix Anal. Appl.*, 35(2):661–683, 2014.

[37] A. Frommer, K. Kahl, T. Lippert, and H. Rittich. Error bounds for the sign function. Technical report, University of Wuppertal, Germany, 2011.

[38] A. Gaul. *Recycling Krylov subspace methods for sequences of linear systems : Analysis and applications.* Phd thesis, TU Berlin, Germany, 2014.

[39] A. Gaul, M. H. Gutknecht, J. Liesen, and R. Nabben. A framework for deflated and augmented Krylov subspace methods. *SIAM J. Matrix Anal. Appl.*, 34(2):495–518, 2013.

[40] A. Gaul and N. Schlömer. Preconditioned recycling Krylov subspace methods for self-adjoint problems. Technical report, TU Berlin, 2015. arXiv:1208.0264.

[41] R. Glowinski and P. Le Tallec. *Augmented Lagrangian and operator-splitting methods in nonlinear mechanics*, volume 9. SIAM, 1989.

[42] G. H. Golub and C. Greif. On solving block-structured indefinite linear systems. *SIAM J. Sci. Comput.*, 24(6):2076–2092, 2003.

[43] G. H. Golub and C. F. Van Loan. *Matrix computations.* The Johns Hopkins University Press, USA, 2013. Fourth edition.

[44] S. Goossens and D. Roose. Ritz and harmonic Ritz values and the convergence of FOM and GMRES. *Numer. Linear Algebra Appl.*, 6(4):281–293, 1999.

[45] P. Gosselet, C. Rey, and J. Pebrel. Total and selective reuse of Krylov subspaces for the resolution of sequences of nonlinear structural problems. *Int J. Numerical Methods in Engineering*, 94(1):60–83, 2013.

[46] S. Gratton, A. Sartenaer, and J. Tshimanga. On a class of limited memory preconditioners for large scale linear systems with multiple right-hand sides. *SIAM J. Opt.*, 21(3):912–935, 2011.

[47] A. Greenbaum, V. Pták, and Z. Strakoš. Any nonincreasing convergence curve is possible for GMRES. *SIAM J. Matrix Anal. Appl.*, 17(3):465–469, 1996.

[48] A. Griewank. Broyden Updating, the Good and the Bad! *Documenta Mathematica*, 301–315, 2012.

[49] M. J. Grote and T. Huckle. Parallel preconditioning with sparse approximate inverses. *SIAM J. Sci. Comput.*, 18(3):838–853, 1997.

[50] M. H. Gutknecht. A brief introduction to Krylov space methods for solving linear systems. In *Frontiers of Computational Science*, pages 53–62. Springer, 2007.

[51] M. H. Gutknecht. Deflated and augmented Krylov subspace methods: A framework for deflated BiCG and related solvers. *SIAM J. Matrix Anal. Appl.*, 35:1444–1466, 2014.

[52] E. V. Haynsworth. Determination of the inertia of a partioned Hermitian matrix. *Linear Algebra Appl.*, 1:73–81, 1968.

[53] V. Hernandez, J. E. Roman, and V. Vidal. SLEPc: a scalable and flexible toolkit for the solution of eigenvalue problems. *ACM Trans. Math. Software*, 31(3):351–362, 2005.

[54] M. R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Research Nat. Bur. Standards*, 49:409–436, 1953.

[55] N. J. Higham. *Functions of Matrices: Theory and Computation*. SIAM, 2008.

[56] T. Z. Huang, G. H. ChenG, and L. Li. New block triangular preconditioners for saddle point linear systems with highly singular (1, 1) blocks. *Mathematical Problems in Engineering*, 13, 2009.

[57] Z. Jia. The convergence of harmonic Ritz values, harmonic Ritz vectors and refined harmonic Ritz vectors. *Math. Comp.*, 74(251):1441–1456, 2005.

[58] C. Keller, N. IM. Gould, and A. J. Wathen. Constraint preconditioning for indefinite linear systems. *SIAM J. Matrix Anal. Appl.*, 21(4):1300–1317, 2000.

[59] C. T. Kelley. *Solving nonlinear equations with Newton's method*, volume 1. SIAM, 2003.

[60] M. Kilmer and E. de Sturler. Recycling subspace information for diffuse optical tomography. *SIAM J. Sci. Comput.*, 27(6):2140–2166, 2006.

[61] A. Klawonn. *Preconditioners for indefinite problems*. Phd thesis, Westfälische Wilhelms-Universität Münster, 1996.

[62] A. Klawonn. Block-triangular preconditioners for saddle point problems with a penalty term. *SIAM J. Sci. Comput.*, 19(1):172–184, 1998.

[63] D. A. Knoll and D. E. Keyes. Jacobian-free Newton-Krylov methods: a survey of approaches and applications. *J. Computational Physics*, 193(2):357–397, 2004.

[64] P. Krzyżanowski. On block preconditioners for saddle point problems with singular or indefinite (1, 1) block. *Numer. Linear Algebra Appl.*, 18(1):123–140, 2011.

[65] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. Research Nat. Bur. Standards*, 45:255–282, 1950.

[66] C. Lanczos. Solution of systems of linear equations by minimized iterations. *J. Research Nat. Bur. Standards*, 49:33–53, 1952.

[67] J. Mandel. Balancing domain decomposition. *Comm. Numer. Methods Engrg.*, 9:233–241, 1993.

[68] J. Mandel and M. Brezina. Balancing domain decomposition for problems with large jumps in coefficients. *Math. Comp.*, 65(216):1387–1401, 1996.

[69] G. Meurant and J. D. Tebbens. The role eigenvalues play in forming GMRES residual norms with non-normal matrices. *Numer. Algorithms*, 68(1):143–165, 2015.

[70] S. Michel-Ponnelle. Modélisation des câbles de précontrainte. Technical Report, EDF R&D, AMA, 2013. R7.01.02.

[71] J. L. Morales and J. Nocedal. Automatic preconditioning by limited memory Quasi-Newton updating. *SIAM J. Opt.*, 10(4):1079–1096, 2000.

[72] R. B. Morgan. Computing interior eigenvalues of large matrices. *Linear Algebra Appl.*, 154:289–309, 1991.

[73] R. B. Morgan and M. Zeng. Harmonic projection methods for large nonsymmetric eigenvalue problems. *Numer. Linear Algebra Appl.*, 5:35–55, 1998.

[74] M. F. Murphy, G. H. Golub, and A. J. Wathen. A note on preconditioning for indefinite linear systems. *SIAM J. Sci. Comput.*, 21(6):1969–1972, 2000.

[75] S. Nash. Newton-type minimization via the Lanczos method. *SIAM J. Numer. Anal.*, 21:770–778, 1984.

[76] J. Nečas and I. Hlavácek. *Mathematical Theory of Elastic and Elastoplastic Bodies.* Elsevier, 1980.

[77] R. A. Nicolaides. Deflation of conjugate gradients with applications to boundary value problems. *SIAM J. Numer. Anal.*, 24(2):355–365, 1987.

[78] J. Nocedal and S. J. Wright. *Numerical Optimization.* Springer Ser. Oper. Res., Springer-Verlag, Berlin, 1999.

[79] Y. Notay. A new analysis of block preconditioners for saddle point problems. *SIAM J. Matrix Anal. Appl.*, 35(1):143–173, 2014.

[80] D. P. O'Leary and A. Yeremin. The linear algebra of block quasi-Newton algorithms. *Linear Algebra Appl.*, 212:153–168, 1994.

[81] M. A. Olshanskii and E. E. Tyrtyshnikov. *Iterative Methods for Linear Systems: Theory and Applications*. SIAM, Philadelphia, USA, 2014.

[82] C. C. Paige, B. N. Parlett, and H. A. van der Vorst. Approximate solutions and eigenvalue bounds from Krylov subspaces. *Numer. Linear Algebra Appl.*, 2:115–134, 1995.

[83] M. Parks, E. de Sturler, G. Mackey, D. Johnson, and S. Maiti. Recycling Krylov subspaces for sequences of linear systems. *SIAM J. Sci. Comput.*, 28(5):1651–1674, 2006.

[84] J. Pellet. Conditions de liaison de corps solide. Technical Report, EDF R&D, AMA, 2011. R3.03.02.

[85] J. Pellet. Dualisation des conditions limites. Technical Report, EDF R&D, AMA, 2011. R3.03.01.

[86] I. Perugia, V. Simoncini, and M. Arioli. Linear algebra methods in a mixed approximation of magnetostatic problems. *SIAM J. Sci. Comput.*, 21(3):1085–1101, 1999.

[87] J. Pestana and A. J. Wathen. Combination preconditioning of saddle point systems for positive definiteness. *Numer. Linear Algebra Appl.*, 2012. doi: 10.1002/nla.1843.

[88] J. M. Proix. Relations de comportement élasto-visco-plastique de Chaboche. Technical Report, EDF R&D, AMA, 2013. R5.03.04.

[89] T. Rees and C. Greif. A preconditioner for linear systems arising from interior point optimization methods. *SIAM J. Sci. Comput.*, 29(5):1992–2007, 2007.

[90] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, USA, 2003.

[91] Y. Saad and M. H. Schultz. GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Statist. Comput.*, 7(3):856–869, 1986.

[92] Y. Saad, M. Yeung, J. Erhel, and F. Guyomarc'h. A deflated version of the conjugate gradient algorithm. *SIAM J. Sci. Comput.*, 21(5):1909–1926, 2000.

[93] R. B. Schnabel. Quasi-Newton methods using multiple secant equations. Technical report, Department of Computer Sciences, University of Colorado, 1983.

[94] N. Sellenet. Calcul élasto-plastique d'un anneau creux soumis à une pression interne. Technical Report, EDF R&D, AMA, 2011. V1.01.245.

[95] J Shewchuk. What is a good linear finite element? Interpolation, conditioning, anisotropy, and quality measures. *University of California at Berkeley*, 73, 2002.

[96] C. Siefert and E. de Sturler. Preconditioners for generalized saddle-point problems. *SIAM J. Numer. Anal.*, 44(3):1275–1296, 2006.

[97] J. Sifuentes. *Preconditioning the integral formulation of the Helmholtz equation via deflation.* PhD thesis, Rice University Houston, 2006.

[98] D. Silvester and A. Wathen. Fast iterative solution of stabilised Stokes systems part II: using general block preconditioners. *SIAM J. Numer. Anal.*, 31(5):1352–1367, 1994.

[99] V. Simoncini and D. B. Szyld. Recent computational developments in Krylov subspace methods for linear systems. *Numer. Linear Algebra Appl.*, 14(1):1–59, 2007.

[100] G. LG. Sleijpen and H. A. van der Vorst. A Jacobi–Davidson iteration method for linear eigenvalue problems. *SIAM Rev.*, 42(2):267–293, 2000.

[101] K. M. Soodhalter, D. B. Szyld, and F. Xue. Krylov subspace recycling for sequences of shifted linear systems. *Appl. Num. Math.*, 81:105–118, 2014.

[102] M. Stoll and A. J. Wathen. Combination preconditioning and the Bramble–Pasciak+ preconditioner. *SIAM J. Matrix Anal. Appl.*, 30(2):582–608, 2008.

[103] K. Stüben. A review of algebraic multigrid. *J. Comput. Appl. Math.*, 128(1):281–309, 2001.

[104] J. M. Tang, S. P. MacLachlan, R. Nabben, and C. Vuik. A comparison of two-level preconditioners based on multigrid and deflation. *SIAM J. Matrix Anal. Appl.*, 31(4):1715–1739, 2010.

[105] N. Tardieu and E. Cheignon. Application d'une méthode de Newton-Krylov en mécanique des solides. In *10e colloque national en calcul des structures*, 2011.

[106] J. D. Tebbens and M. Tuma. Efficient preconditioning of sequences of nonsymmetric linear systems. *SIAM J. Sci. Comput.*, 29(5):1918–1941, 2007.

[107] A. Toselli and O. Widlund. *Domain decomposition methods: algorithms and theory*, volume 3. Springer, 2005.

[108] U. Trottenberg, C. W. Oosterlee, and A. Schuller. *Multigrid*. Academic press, 2000.

[109] J. Tshimanga. *On a class of limited memory preconditioners for large-scale nonlinear least-squares problems*. Phd thesis, University of Namur, Belgium, 2007.

[110] H. A. van der Vorst. *Iterative Krylov methods for large linear systems*, volume 13. Cambridge University Press, 2003.

[111] S. C. Eisenstat H. F. Walker. Choosing the forcing terms in an inexact Newton method. *SIAM J. Sci. Comput.*, 17(1):16–32, 1996.

[112] S. Wang, E. de Sturler, and G. Paulino. Large-scale topology optimization using preconditioned Krylov subspace methods with recycling. *Int J. Numerical Methods in Engineering*, 69(12):2441–2468, 2007.

[113] U. M. Yang. *A family of preconditioned iterative solvers for sparse linear systems*. PhD thesis, University of Illinois at Urbana-Champaign, 1995.

[114] U. M. Yang and K. A. Gallivan. A new family of preconditioned iterative solvers for nonsymmetric linear systems. *Appl. Num. Math.*, 19(3):287–317, 1995.

Résumé

La thèse a pour objectif le développement de méthodes performantes pour la résolution de problèmes non-linéaires en mécanique des solides. Il est coutume d'utiliser une méthode de type Newton qui conduit à la résolution d'une séquence de systèmes linéaires. De plus, la prise en compte des relations linéaires imposées à l'aide de multiplicateurs de Lagrange confère aux matrices une structure de point-selle.
Dans un cadre plus général, nous proposons, étudions et illustrons deux classes d'enrichissement de préconditioneurs (limited memory preconditioners) pour la résolution de séquences de systèmes linéaires par une méthode de Krylov. La première est une extension au cas symétrique indéfini d'une méthode existante, développée initialement dans le cadre symétrique défini positif. La seconde est plus générale dans le sens où elle s'applique aux systèmes non symétriques. Ces deux familles peuvent être interprétées comme des variantes par blocs de formules de mise à jour utilisées dans différentes méthodes d'optimisation. Ces techniques ont été développées dans le logiciel de mécanique des solides Code_Aster (dans un environnement parallèle distribué via la bibliothèque PETSc) et sont illustrées sur plusieurs études industrielles. Les gains obtenus en terme de coût de calcul sont significatifs (jusqu'à 50%), pour un surcoût mémoire négligeable.

Mots-clefs : Mécanique des solides, Itérations de Newton, Systèmes point selle, Préconditionneurs à mémoire limitée, Vecteurs de Ritz (harmonique)

Abstract

The thesis aims at developing efficient numerical methods to solve nonlinear problems arising in solid mechanics. In this field, Newton methods are currently used, requiring the solution of a sequence of linear systems. Furthermore, the imposed linear relations are dualized with the Lagrange multiplier method, leading to matrices with a saddle point structure.
In a more general framework, we propose two classes of preconditioners (named limited memory preconditioners) to solve sequences of linear systems with a Krylov subspace method. The first class is based on an extension of a method initially developed for symmetric positive definite matrices to the symmetric indefinite case. The second class handles the more general case related to nonsymmetric matrices. Both families can be interpreted as block variants of updating formulas used in numerical optimization. They have been implemented into the Code_Aster solid mechanics software (in a parallel distributed environment using the PETSc library). These new preconditioning strategies are illustrated on several industrial applications. We obtain significant gains in computational cost (up to 50%) at a marginal overcost in memory.

Keywords: Solid mechanics, Newton iterations, Saddle point systems, Limited memory preconditioners, (Harmonic) Ritz vectors